Abstract

The main aim of this project is to research the possible solutions and implement the related algorithms, which can realize the effect of shallow depth of field on an image. The expected program can help people take images with shallow depth of field by digital camera. For instance, the emphasis of a photograph is on the persons rather than their background such as buildings. I intend to make the foreground and background blurred. This is a comprehensive and interdisciplinary project. There are three algorithms which are from computer vision, computer graph, and photography. The first algorithm derived from depth from defocus. The second is base on multiple rendering and update a new ray tracing algorithms. The third algorithm is discovered by camera imaging features.

- Do research photography and find out the factors that affect the depth of field. Compare and analysis the results, and discovery imaging law. See page 9-13
- Description methods respectively and found the most suitable for using on the camera. See page19-25
- Update a new rendering algorithm to suit rendering based on bitmap. See page 36-38 .
- The experiment of Multiple Rendering algorithmSee page 41-42.
- The experiment of Precise Rendering algorithm See page 43-46.
- The experiment of Vignetting PhenomenonSee page 46-48

Acknowledgements

Like so many tasks, it would not be possible without the help, advice and encouragement of many people. Firstly, I would like to give thanks to my supervisor Erik Reinhard. This project is just from my funny idea. He gave me the direction to solve some problems. And I also would like to thank my friends who give me so much support and help.

Finally, I will give my deeply appreciate to my parents. They give me the educational environment, and always supported my choice, when I was very young. I also thank them for they give the financial support and this opportunity.

Content

Introduction....4
1 Aims and Objectives....5
2 Background and Previous Work....6
2.1 Introduction....6
2.2 Depth of Field....9
2.2.1 Depth of field formula....9
2.2.2 Fore and back depth of field....11
2.3 Obtaining depth information....14
2.3.1 Background and significance....14
2.3.2 Measurement Introduction....14
2.4 Depth from defocus....17
2.4.1 Development of Depth from defocus....17
2.4.2 Algorithms Introduction....19
2.5 Rendering algorithm....26
2.5.1 Background....26
2.5.2 Depth of Field Theory....27
2.6 Previous work....28
2.6.1 Based on optical simulation....28
2.6.2 Using image processing software....30
2.6.3 Pro Focus....31
3 Algorithm and design....34
3.1 Overview....34
3.2 Seeking depth algorithm and design....34
3.3 Rendering image algorithm and design....36
3.4 Create vignette algorithm and design....39
4 Experiments....40
4.1 Overview....40
4.2 Results and Comparison....40
4.3 Simulation of phenomena....48
5 Conclusion....51
6 FUTURE WORK....54

Introduction

With the development of computing power, computer graphic algorithms have become more complex and perfected. In recent movies and games, depictions of focus, optical aberration, and depth of field (DOF) can be increasingly found, which has rarely been seen in the past ten years. Due to the development of GPU computing power, more real and perfect images have been produced. Meanwhile, at the same time, there has been an increase in the popularity of digital cameras, where people can take photos anywhere at any time. Compared with film cameras, digital cameras are much more convenient. However, the hardware for digital cameras causes the digital images to be without some artistic factors. The most obvious factor is that the digital cameras cannot shoot images in the same way as film cameras do with shallow DOF. Furthermore, there are some situations where controlling DOF is important, such as the most common portrait photography. Good portrait photography should use large aperture in order to achieve a shallow DOF, in other words, there is not a lot of sharp foreground and background which can distract attention away from the person.

The main objective of this project is to research the possible solutions and implement the related algorithms, which can realize the effect of shallow DOF on an image. The expected program can help people take images with shallow DOF by digital camera. For instance, the emphasis of a photograph is on the persons rather than their background such as buildings. I intend to make the foreground and background blurred. In other words, the shape outside the focus appears more blurred than in the original photograph.

This is a comprehensive and interdisciplinary project. There are three algorithms which are from computer vision, computer graph, and photography. The first algorithm derived from depth from defocus. The second is base on multiple rendering and update a new ray tracing algorithms. The third algorithm is discovered by camera imaging features.

List algorithms have different methods for depth from defocus. Introducing the background and the derivation of the formula, these lay the foundation for the new algorithm. Comparing the test result is from the new algorithm with existing algorithm.

1 Aims and Objectives

Control over depth of field is an important artistic tool that can be used to emphasize the subject of a photograph. Without depth of field, everything appears in sharp focus, leading to an unnatural, overly crisp appearance. The aim of this project is to develop an application which can simulate a more shallow depth of field.

The depth of field effect has been simulated in computer graphics in accordance with the same effect as found in real camera lenses.

The objectives to meet this aim can be broken down as follows:

1. Research photography to find out the factors that affect the depth of field. When compare and analysis the results, this work can help us discovery the cause and determine whether the results are correct.
2. Study and seeking ranging method in order to recovery depth. Description methods respectively and found the most suitable for using on the camera.
3. Choose a suitable algorithm in depth from defocus, and study the difference between this algorithm and others.
4. According to the parameters in selection algorithm, shooting many different kinds of related photos.
5. Implement the algorithm and test.
6. Study and find out a suitable algorithm about rendering. Because the rendering data type is different from the common rendering model, so use two algorithms to finish this part. One is existing, another need update original.
7. Study different types of fuzzy filter algorithms and results, in order to update a new algorithm for rendering base on an image with depth of field.
8. Design and Implement the update rendering algorithms.
9. Comparing final output images with the goal of images, find out the differences.
10. Based on the result of 8, try to improve the effect.
11. All the implementation will be done with C++ programming.

Organization of this dissertation

This dissertation will first introduce some background knowledge and related knowledge. Analysis some result from previous work done by the others. Introduction of algorithms used and code structures in my implementation will be given in Chapter 3. Chapter 4 will show and compare the result with goal of effect images. Chapter 5 will be conclusion and evaluation about my implementation. There will be some future work in Chapter 6

2 Background and Previous Work

2.1 Introduction

Photography is unlike painting where objects can be selected to be included in the picture or not. In photography, all the objects are within the scene, whether or not you want them. Too many objects which are not needed remain in the picture, and they will affect the performance of the main subject. For example, athletes are running, but there are too many viewers around the athletes. No matter how you frame the photo, the audience will always be taken into the screen. If the audience is as clear as the athletes in the photo, the athletes will not be performing well. If a photographer wants to highlight the athletes, the solution is to make the audience blurred as a background to the image. To do so requires a telephoto lens with the aperture open to the maximum. These can make the photo have a slight depth of field, with the background blurred. Thus, the depth of field in the picture is very important.

With the development of photographic equipment, people interested in photograph strive to their improve skills in order to get a high quality bokeh [1] and increase the slight depth of field. In photography, bokeh is the aesthetic quality of the blur[1]. The lens renders some parts of image which are out of focus as more smooth and more beautiful. Differences in lens aberrations and aperture cause some lens designs to blur the photo. This can be pleasing to the eye, while others produce blurring that is unpleasant or distracting. Good or bad bokeh can also make a picture good or bad. Bokeh occurs only in the parts of the scene which lie out-of-focus. However, when photographers deliberately use a shallow depth of field technique to create images with prominent out-of-focus area, bokeh will produce a picture of the soul.

Figure 2.1: Bad bokeh and Good bokeh (from left to right) [2]

In order to get a good bokeh, manufacturers are doing continuous efforts. Increasing the number of leaves in lenses, using round circle leaves, producing special mirrors, all of this achieves better blur effects, and reduces the appearance of the bokeh second linear. The photos in Figure2.2 give the examples.

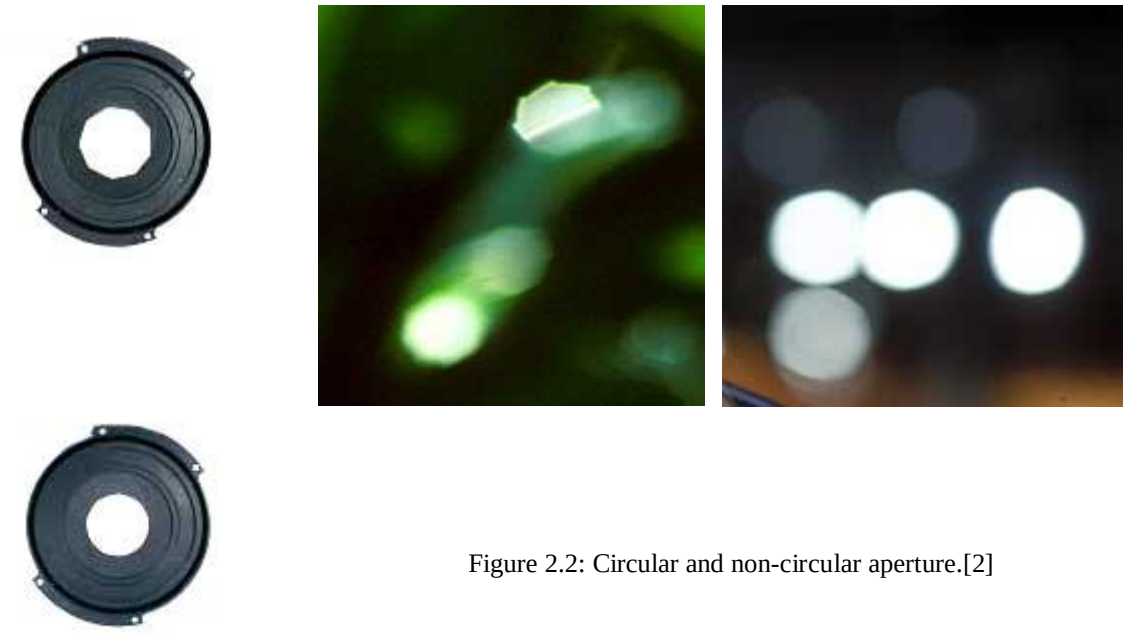

Figure 2.2: Circular and non-circular aperture.[2]

The photo on the left in Figure 2.1 shows a bad bokeh with second linear phenomenon, the objects in background are defocusing with two or more contour lines. This phenomenon is easy to disturb the main objects. The photo on the right in Figure 2.1 shows a good bokeh. In addition to the focus, the entire picture is spread, fuzzy and has no remains of the edge; this is the best-focus imaging. For an ordinary refractive lens, there are two factors which impact the image appearing out-of-focus; one is the nature of the lens design while the other is the shape of aperture. Most lenses are at maximum aperture when the aperture is circular. At this time, an out-of-focus lens is designed to represent nature. When the aperture becomes smaller, photographs show that the out-of-focus is affected by the lens design and the shape of the aperture. Some full-aperture lens can show good bokeh, but in the contraction of the aperture, due to the non-circular aperture, which will destroy the image in out-of-focus. This is why many manufacturers use a circular aperture in successive reasons. When the circular aperture is in the contract two steps, it will still be able to take a more perfect circle when out-of-focus.

.

Figure 2.3: Out-of-focus description by different lenses.[2]

The photo on the left in Figure 2.3 shows spherical aberration caused by out-of-focus, although proliferation, but in the remaining part of the edge. The middle photo shows a bad out-of-focus image as the edge is too sharp. The right photo is naturally out-of-focus with no remains of the edge. This is the best one. Using this kind of lens can create good bokeh.

Meanwhile, some companies add some functionality to the camera body. Some companies design a number of expensive large aperture lens. These are intended to make the image in focus become sharp and clear while objects which are out-of-focus look very smooth, without disrupting the line of sight.

Now, with the popularity of digital cameras, photography is no longer the work for only professional photographers. More and more people can shoot great photos which are at the same level of professional photographers. The invention of digital cameras makes photography even more popular, but due to the reasons including design and production costs, they do not have original film cameras' depth of field. This makes the pictures lack artistic sense. In the past four or five years, some image processing software has become able to produce the shallow depth of field effect. They will be described in later of this chapter.

This section will describe the depth of field, looking for factors that affect the depth of field and will also introduce some knowledge used in Chapter 3, such as how to measure distance and use the distance to render a new image.

2.2 Depth of Field

In an image, when a lens focuses on a plane, with the distance changing the clarity of the object changes, this is the depth of field. There are four factors governing the depth of field.

- The focal length of the lens
- The aperture setting
- The focus-distance setting
- The circle of confusion

If a person changes the values of any variable, the depth of field will change. This is in strict accordance with the mathematical laws of optics. In addition to these four variables, there is a lesser-known but very important hidden factor. This is the size of the film, which today is sensor such as CCD (Charge Coupled Device) and CMOS (Complementary Metal-Oxide Semiconductor). Film area compared to a digital camera sensor, is 6 to 12 times larger. Even if the size of the sensor in non full frame digital single lens reflex (DSLR) is only half the size of the film, due to the hardware, it is difficult to shoot in the same way as a film camera shooting an image with a shallow depth of field.

2.2.1 Depth of field formula

We know, just make sure the depth of field is close to the boundaries, and can determine the depth of field. u_{near} is near the boundary, u_{far} is far from the boundary, u is Object Distance, f is Lens focal length, F is aperture size.

$$u_{near} = \frac{f^2 u}{f^2 + (u - f) Fd} \quad (1)$$

$$u_{far} = \frac{f^2 u}{f^2 - (u - f) Fd} \quad (2)$$

If calculating the depth of field, the amount of substitution can calculate the result. u_{depth} is the depth of field.

$$u_{depth} = u_{far} - u_{near} \quad (3)$$

For observation, it will lead x into formulate (3).

$$x = \frac{u - f}{f^2} Fd \tag{4}$$

So it can produce:

$$u_{depth} = u_{far} - u_{near} = u\left(\frac{1}{1 - x} - \frac{1}{1 + x}\right)$$
$$u_{depth} = \frac{2ux}{1 - x^2}. \tag{5}$$

Because the far boundary (u_{far}) is infinite, the range x is $0 < x < 1$. By (4) shows that, when using long focal length lens, with a large aperture for close shoot photo, x is much smaller than 1 the approximate conditions are easily met.

When x is far less than 1, in (5) x^2items can be ignored, the depth of field equation can be simplified to:

$$u_{depth} = 2ux \tag{6}$$

When shooting, the focus distance is at least 10 times the lens focal length. So, this can be estimated:

$$u_{depth} = \frac{2u^2}{f^2} Fd \tag{7}$$

Through analysis of this formula, three points are revealed:

- First, the depth of field and the value of aperture are proportional.
- Second, the depth of field and the square of shooting distance are proportional.
- Third, the depth of field and the square of the lens focal length are inversely proportional.

These results will check the results of rendering in chapter 4.

2.2.2 Fore and back depth of field

Depth of field in focus from the formula gives a clear image when a known object is in distance range. However, in the actual shooting, we need to know objects in the case of distribution to choose the right focus distance. This requires an understanding of the relationships between fore depth of field and back depth of field.

Fore depth of field is the distance measured from the near boundary to focal plane.

$$u_{fore} = u - u_{near} \tag{8}$$

$$u_{fore} = \frac{2ux}{1 - x^2} \tag{9}$$

Back depth of field is the distance measured from the local plane to back depth of field.

$$u_{back} = u_{far} - u \tag{10}$$

$$u_{back} = \frac{2ux}{1 - x^2} \tag{11}$$

When x is far less than 1 second, the time may be ignored:

$$u_{fore} = u_{back} = 2ux \tag{12}$$

		F= 2	F= 4	F= 8	F= 16
u = 1m	u_{near}	0.963	0.929	0.868	0.767
	u_{far}	1. 040	1. 082	1. 179	1. 437
	u_{depth}	0. 077	0. 153	0. 311	0. 670
u = 2m	u_{near}	1. 855	1. 730	1. 524	1. 232
	u_{far}	2. 169	2. 370	2. 907	5. 319
	u_{depth}	0. 314	0. 640	1. 383	4. 087
u = 4m	u_{near}	3. 454	3. 040	2. 451	1. 767
	u_{far}	4. 751	5. 848	10. 870	
	u_{depth}	1. 297	2. 808	8. 419	
u = 8m	u_{near}	6. 070	4. 890	3. 521	2. 257
	u_{far}	11. 730	21. 978		
	u_{depth}	5. 66	17. 088		

Figure 2.4: Depth of field table. The lens focal length is 50mm. d=0.05mm

		F= 2	F= 4	F= 8	F= 16
u = 1m	u_{near}	0. 991	0. 982	0. 965	0. 933
	u_{far}	1. 009	1. 018	1. 037	1. 078
	u_{depth}	0. 018	0. 036	0. 072	0. 145
u = 2m	u_{near}	1. 963	1. 927	1. 859	1. 736
	u_{far}	2. 039	2. 079	2. 165	2. 358
	u_{depth}	0. 076	0. 152	0. 306	0. 622
u = 4m	u_{near}	3. 850	3. 711	3. 460	3. 049
	u_{far}	4. 162	4. 338	4. 739	5. 814
	u_{depth}	0. 312	0. 627	1. 279	2. 765
u = 8m	u_{near}	7. 414	6. 908	6. 079	4. 902
	u_{far}	8. 686	9. 501	11. 696	21. 739
	u_{depth}	1. 272	2. 593	5. 617	16. 837

Figure 2.5: Depth of field table. The lens focal length is 100mm. d=0.05mm

		F= 2	F= 4	F= 8	F= 16
u = 1m	u_{near}	0. 998	0. 996	0. 992	0. 984
	u_{far}	1. 002	1. 004	1. 008	1. 016
	u_{depth}	0. 004	0. 008	0. 016	0. 032
u = 2m	u_{near}	1. 991	1. 982	1. 965	1. 931
	u_{far}	2. 009	2. 018	2. 037	2. 075
	u_{depth}	0. 018	0. 036	0. 072	0. 144
u = 4m	u_{near}	3. 962	3. 925	3. 854	3. 717
	u_{far}	4. 038	4. 077	4. 158	4. 329
	u_{depth}	0. 072	0. 152	0. 304	0. 612
u = 8m	u_{near}	7. 847	7. 700	7. 421	6. 920
	u_{far}	8. 159	8. 325	8. 677	9. 479
	u_{depth}	0. 312	0. 625	1. 265	2. 559

Figure 2.6: Depth of field table. The lens focal length is 200mm. d=0.05mm

Analysis of Figures 2.4 to 2.6, the data can be drawn:

- When the focal length of the lens is 200mm, the aperture is less than F8 and the object distance is less than 2m, or the aperture is less than F4 and the object distance is less than 4m, fore depth of the field is the same as back depth of the field.

- When the focal length of the lens is 100mm, the aperture is less than F8 and the object distance is less than 1m, or the aperture is less than F4 and the object distance is less than 2m, fore depth of the field is the same as the back depth of the field.

- When the focal length of the lens is 50mm, the aperture is less than F2 and the object distance is less than 1m, the fore depth of the field is the same as the back depth of the field.

Therefore, when using a large camera shooting distance and small aperture with the use of a short focal length lens, the back depth of the field will be much larger than the fore depth of the field.

2.3 Obtaining depth information.

2.3.1 Background and significance.

Computer vision technology is a branch of artificial intelligence, and is research based on digital image processing. It can get visual information from cognitive processes. It is the study of visual information processing theory of computation and uses expression and calculation methods, including image feature extraction, camera calibration, stereo vision[17], sports vision, or image sequence analysis, three-dimensional objects by the gray shape recovery methods[6], object modeling and recognition through image analysis methods.

Recovering three-dimensional information of objects in images is a very important part of computer vision. In estimating the target object, the three-dimensional reconstruction[17] of depth information is a critical step in the process.

In addition, there are many objects in the depth of the image restoration methods, for example: stereo vision, structure from motion, and depth from defocus[7]. These methods have been proven in the literature, these techniques can be applied to the reconstruction of three-dimensional space, image object recognition and many other aspects and bring us much useful help.

2.3.2 Measurement Introduction

Measured according to the need for human-controlled environments, distance measurement methods can be divided into two. One is the active sensing method; another is the passive sensing method.

Active sensing methods

Active sensing method needs to send artificial light to irradiate objects, such as lasers which are texture light. By analyzing the deformation texture of the object of reflected

light or direct light propagation time measurement is used to determine the object distance. The object is measured by whether the contacts and the active sensing method is divided into the contact distance measurement sensing method or non-contact distance measurement sensing method.

The contact distance measurement method uses a very sensitive probe along the surface with point by point measurements. It needs a mechanical movement that sweeps along the surface. Therefore, this measure is very expensive and time consuming, for a soft object, the method is not applicable.

Active non-contact distance measurement methods can be divided into, optical distance measurement method and non-optical active distance measurement method. Non-optical active distance methods are: Microwave radar and sonar ranging distance. Optical active ranging methods are: Structure light field method and time of flight methods.

Other methods include the flying time modulated laser beam which works by measuring the beam from the object and the sensor time to calculate the distance between object and sensor. This method is suitable for measuring distance and less precise objects. Objects using a structured light field method of measuring the distance from the principle, and are based on triangulation principle, by measuring the deformation of the pattern projected by the object distance information, the measurement accuracy. The time of flight method requires modulation of the laser beam and is achieved by measuring the beam from the object and the sensor to calculate the distance between the object and the sensor. This method is suitable for measuring distant and less precise objects. The structure light measurement method is used for object distance and is based on the triangulation principle. Projection design is achieved by measuring the deformation by the object distance information. This method is highly accurate.

Passive sensing methods

Passive ranging detects and analyzes the light radiation from a natural object to determine the object's distance. There are some typical passive ranging methods: stereo vision[17], structure from motion[18], monocular distance[19].

Stereo vision involves two or more cameras capturing images and matching their features. If the object feature points in a similar position in the two images, which shows the feature points of two cameras at the intersection of the axis, the objects in the other characteristics in different positions determine the feature point’s position.

The basic principle of stereo vision is observed from two or more of the same scene points of view in order to obtain images from a different perspective. This is achieved through the triangulation principle and by calculating the position deviation between the image pixels, and then by obtaining three-dimensional scene information. The three-dimensional process and human visual perception process is similar. A complete stereo vision system can generally be divided into: image acquisition, camera calibration, feature extraction, stereo matching, depth determination and interpolation of several parts.

The structure from motion method is the method of trading time for space. An imaging sensor is required to obtain a series of images at different times and different spatial locations. Based on a two-dimensional image sequence of time and space change, it can extract information from the surface of the object. Structure from motion methods and stereo vision ranging methods are similar in image processing. The key technology is also found in the corresponding point in the different images.

Both stereo vision and structure from motion methods need to find corresponding features in images, and calculate the displacement between them. Extraction of object features is a very time-consuming and complex task. These two methods are limited by the block problem. If an object can be visible in images, but not visible in other images, then the corresponding features cannot be found. Thus, the depth of the object cannot be calculated. This leads to them being used to measure depth and is very difficult.

There are some methods in monocular ranging methods, such as depth from focus and depth from defocus. [9] The depth from focus method uses different optical parameters for shooting. To find the clearest image in a series of objects, the imaging principles of geometrical optics are needed to calculate the object depth. This method can obtain more accurate dense depth information of objects. However, it requires different optical parameters and a large number of images taken by the objects, thus it is difficult to measure in real time.

Depth from defocus is about obtaining distance information from the changes of defocus. [7] According to the principle that is the greater degree of defocus the image will be more blurred. It will use two or three images to determine the parameters of focal point spread. This calculation is based on the defocus parameters relationship with object distance.

Depth from defocus is different from stereo vision and the structure from motion method as this method does not need to find the corresponding features in different images. Because the image acquisition process maintains the same direction, it eliminates the problem of objects being blocked in stereo vision. Depth from the defocus method does not require a large number of images taken as the depth from focus method.

Because the image acquisition process maintains the same direction, it eliminates the stereo vision and motion visual occlusion. Compared with the focus ranging method, it does not take a lot of image manipulation operations on the local image into consideration, although the method is simple, fast, and can measure the depth of high-density objects. Operation and calculations are carried out on the local image. This method is simple and fast, it can be used to measure the depth of high-density objects.

2.4 Depth from defocus

2.4.1 Development of Depth from defocus

There are many researchers which study the image blurring problem when calculating the depth of the object. The following paragraphs look at the work in this field review.

In 1987, Pentland[5] for objects with step edges was the first time an image was blurred by calculating the distance between the object methods. Assuming the imaging system point spread function of Gaussian distribution, Laplacian operator with step-edge image of the fuzzy measure the degree of defocus used to calculate the diffusion parameters of the object distance.

In 1988, Subbarao and Gurumoorthy discovered the diffusion parameters of the line spread function of the distance that an object calculates the object depth of the step edge. Their calculation, the abolition of the point spread function of Gaussian function limitation, requires only a point spread function to have a circular symmetry. The Pentland method[5] can be applied to a wide range of optical systems.

Pentland's idea was based on different optical parameters obtained by two aerial image depth pieces of information. It requires shooting an object image with a pinhole aperture column to make the objects within the image clear throughout, while another image is used in a large aperture to shoot. Then, these two objects correspond with the sub-image in many blocks, calculating the corresponding sub-block in one or more bands within the frequency band signal energy, through the corresponding signal energy to find out the location of the object distance. The biggest drawback of this method is to measure the pinhole aperture used, resulting in very little energy in the camera system, so that the captured images have a lot of noise which seriously affects the measurement accuracy.

In the same year, Subbarao's methods in the study based on the Pentland removed restrictions on shooting conditions by assuming that the system point spread function to meet the Gaussian distribution, optical system parameters to find the infinitesimal change and limited change, defocusing diffusion parameters and the optical system parameters relationship. Subsequently, the method has been extended to a non-Gaussian distribution of scattered point spread function.

In the early 90s, in order to improve the accuracy of defocus ranging method, Ens and Lawrence proposed a defocus distance of the iterative matrix method: looking for two different optical parameters of image defocus convolution transformation matrix. Convolution represents the transformation matrix between the two defocus image point spread function by calculating the point spread function of the diffusion parameter to estimate the depth distribution of the object.

In 1992, Subbarao and Wei illustrated the point spread function of the diffusion parameters for the two images, one-dimensional Fourier transform of the ratio of the parameters. In the frequency domain, the use of two images of the spectra to defocus the depth of the object, the method to remove the Pentland pinhole images taken defocus point spread function limitations and constraints of the model were first identified. The spatial resolution of 128 × 128 pixels, and the mean square error of depth measurements was 3.5%.

In 1994, Xiong and Shafer made the vector filter to compensate for the defocus transfer function changes with frequency due to non-uniformity. In the frequency domain, the dominant in the two images is used to calculate the dominant frequency of the depth of the object.

In 1995, Klarquist [16] proposed the use of multiple images for defocus distance of maximum likelihood estimation. This approach defocused all previous measurements as being not the same and can be thought of as a new defocus measurement. Unfortunately, their method requires not only finding the best image point position, but also because of the possible increase in the number of unknowns to estimate the value of the more vulnerable to noise.

In 1995, Watanabe and Nayar used Lapalcian defocus distance but the Gaussian operator could not be considered accurate because the depth of measurement is the band width of the filtering operator which is very wide. To compensate for this shortcoming, the broadband rational filtering operator to acquire accurate depth of the object methods was designed. This method can achieve the spatial resolution of 7 × 7 and is a passive method of defocusing the ranging relatively high spatial resolution.

In 1995, Raj agopalan and Chaudhuri divided the image into multiple sub-blocks, in which the overlap between adjacent sub-blocks, and then change the region for the displacement calculation, because the point spread function PSF to solve the image of

the surrounding area is caused by overlapping. In 1998, they assumed that the image is the premise of the process from the return to Caramer-Rao lower bound to calculate the optimal defocus distance of the camera model parameters.

In 1999, Schechner and Kiryati proposed the use of sensitivity analysis using the cylindrically-shaped point spread function PSF to estimation error and the relationship between different frequency components. They proposed defocusing the image based on the estimated distance method as being a more accurate orientation. However, they were only able to understand how to adjust the model parameters to estimate the relative error to a minimum and they did not mention how the parameters affect the accuracy of DFD.

In addition, because they used the PSF for the calculation of defocused images more seriously, its results cannot be applied to slightly defocus the image. The Raj agopalan and the PSF used Chaudhuri although at a slightly higher level than for image distortion. However, they assumed self-image as the return of AR (Autoregressive) model generated, so they calculated the CRLB through its association with the image content. Plus, they only made a set of computer simulation data and it is difficult to use other derived data accordingly.

2.4.2 Algorithms Introduction

Recovering depth base on two images

Simple thin lens defocused imaging model as depicted in Figure 2.7. P is an object.' u' is the object distance. 'f' is Focal Length .' v' is the distance of Image Focus. The relation between f, v and u is expressed by the well-known lens formula:

$$1 / u + 1 / v = 1 / f \qquad (13)$$

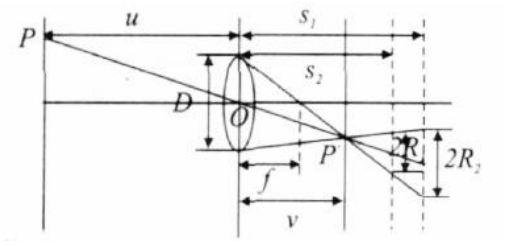

Figure2.7: Show the monocular vision imaging.

P point through a lens into the image of P 'is like a focal point for the plane, then get the image is focused. If the image plane at this time not in the focus plane position, P point of imaging will not focus to a point. It produces a fuzzy round spot. When the image plane move to the optical axis position s1 or s2, P point of image will spread into the circle of confusion. The radiuses are R1 and R2. At this time the image was taken by defocus. From (13) find the fuzzy round radius: Ri (i = 1, 2). D is the Aperture Diameter. D = f /F, F is the lens aperture. Camera parameters is η(η =(f, F, s)).

$$Ri = siD /2 (1 / f - 1 / u - 1 / si) \quad (14)$$

Suppose P points in the three-dimensional space coordinates x, y is its corresponding image plane coordinates. Capture an image using the real aperture optical system parameters for η, the defocused imaging process can be represented as the following integral form:

$$I (y,\eta) = \int h\eta (y, x) r(x) dx \quad (15)$$

r (x) is the scene surface emissivity; hη (y, x) is the imaging core. Camera lens point spread function (point spread function, PSF).The imaging devices is relevant with geometry. $\int\Omega$ hη (y, x) dx = 1,Ω represent the surface of any scene. Although in the same scene, the fuzzy degree in images will be different because of camera parameters. (σ represent the image blurring.) So it can derive the camera parameter information from the fuzzy degree.

$$\sigma = kR \quad (16)$$

Image preprocessing

By using different camera parameters to taken images, the brightness and magnification will be different. If we just change the aperture value and take two images, we can get two defocus images with little change in magnification. We need standardize the brightness of the images. In this step this algorithm is much easier than some algorithm which is get defocus images by changing focal plane. We do not need standardize the magnification. Just use the image average gray value divided by the corresponding image pixel gray value of each, everything is done.

$$I1 (n) = - 3 [5n2 - (3N2 + 3N - 1)] / [(2N - 1) (2N + 1) (2N + 3)] \quad (17)$$

Where: n = - N, - N + 1, ⋯, - 1, 0, 1, ⋯, N - 1, N.

Camera parameters

In the defocusing imaging process, the actual focal length f and aperture F imaging is read from the camera. But the image distance s it takes to get a special calibration. The following I will give more detail about how to use the two defocusing the image information to calibrate the focal length f of the image distances.

A flat object placed in the center of u0 from the camera position. Camera focal length while maintaining the same situation, adjust the aperture to the F1 and F2 index shot two defocus images g1 and g2. η1 =(f1 , F1 , s1) and η2 =(f2 , F2 , s2). To mark the sake of simplicity, the processed image is still recorded as g1 and g2. After standardization of the g2, gi (i = 1, 2) put the Ri into formula (14)

$$R\,i = s2\,fi\,/2Fi\,(1\,/\,fi - 1\,/\,u0 - 1\,/\,si) \tag{18}$$

We can get mi:

$$m\,i = -\,fi\,s2\,/2Fi,\ ci = -\,m\,i\,(1\,/\,fi - 1\,/\,si) \tag{19}$$

$$\sigma i = kR\,i = kFi\,s2\,/2Fi\,(1\,/\,fi - 1\,/\,u0 - 1\,/\,si) = km\,i\,/\,u0 + kci \tag{20}$$

By STM algorithm [3], as the image of cubic polynomial, two images of fuzzy parameter σ 1 and σ 2:

$$\sigma_1^2 - \sigma_2^2 = 4\,(g_1 - g_2)\,/\nabla^2 g \tag{21}$$

∇^2 is Laplacian Operator

$$\nabla^2 g = (\nabla^2 g_1 + \nabla^2 g_2)\,/2 \tag{22}$$

$$L2\,(n) = 30\,(3n2 - N\,(N + 1))\,/N\,(N + 1)\,(2N - 1)\,(2N + 1)\,(2N + 3) \tag{23}$$

The results of the two convolution sum, we get the image Laplacian transform. n = - N, - N + 1, ⋯, - 1, 0, 1, ⋯, N - 1, N.
By the formula (20) may be the relationship between σ1 and σ 2:

$$\sigma 1 = a\sigma 2 + b \tag{24}$$

$$a = m1\,/m2,\ b = k\,(c1 - ac2) \tag{25}$$

As two images taken with the same focal length and image distance, the only change is aperture value, f2 = f, s1 = s2 = s, F1 ≠F2.

$$a = F2 / F1,\ b = 0 \qquad (26)$$

$$\sigma 2 = \pm G\sqrt{(a2 - 1)},\ \sigma 1 = a\sigma 2 \qquad (27)$$

Here are two sets of solutions σ2. In order to obtain only the correct solution, we use the following method to determine the sign of σ2. Adjust the camera's focal length of the image tends to clear, set a focal length f0, If $f \leq f0$, then take a positive sign σ2; or σ2 take a negative sign.

$$si = ((2Fi\ \sigma i) / kfi + 1) / (1 / fi - 1 / u0) \qquad (28)$$

In many papers written about depth of focus, the camera parameters are known quantities, and focus on research method. In fact the determination of camera parameters are complex and difficult to achieve. This algorithm can be run without precise calibration of instruments when trying to meet the function.

Recovering depth based on one image

Another algorithm is about recovering depth from defocus with one image more than one. I will show this algorithm, because it is simple operation, and the most important factor is that we do not need to know any information about the image which will be used for recovering depth. In some algorithm we need know the slope of the edge of the object, or the edge of the position of the image in focus. Although this kind of algorithm calculates results which will be more secure and more stable but, it restricts the use of area. This algorithm uses the local line to recover depth.

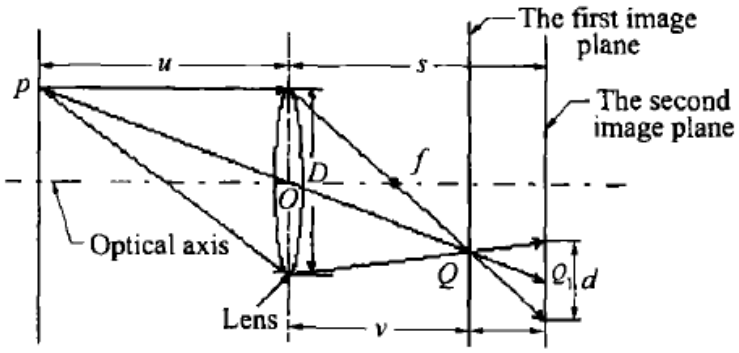

Figure 2.8: Shows the monocular vision imaging.

P is a point on the surface. Q is the point of focus imaging. Q1is the defocus image dispersion circle's center. f is the lens focal length, u is the object distance. v is the image distance when P point has focused image. s is the image distance when P point has defocus image.

As mentioned in the previous algorithm, When the P point position unchanged, imaging plane position moves on both sides of focal plane. According to the formula (29), we can get: P point imaging is no longer as a point, become a round spot. When the image plane at image plane 2, the circular spot diameter d:

$$d = sD\left(\frac{1}{f} - \frac{1}{u} - \frac{1}{v}\right) \tag{29}$$

D is the lens aperture. Suppose σ is the spread function of expansion of 2-dimensional Gaussian distribution parameters. It is relationship with the circular spot diameter that can be expressed as: d = kσ. Where k is a constant need calibration. D = f / F. F is the lens aperture.

$$u = \frac{sf}{s - f - Fk\sigma} \tag{30}$$

If the camera's optical system parameters are known, (such as F, s, f) and σ, and k, we can obtain u, s and f can obtained by the calibration process.

Local line method

In practice, isolated points on the object do not exist. This is due to the points on the surface continuity, and the object edge blurring in the image is very clear. Hence, we can use local line to measure depth.

Suppose there is a step edge, and this edge is a local straight edge. Linear equation is:

$$x = e(y) = hl\ y + h2 \tag{31}$$

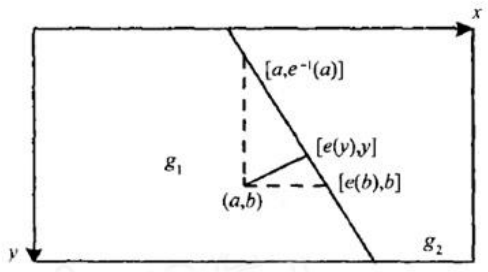

Figure2.9: Edges and adjacent points of a random line.

The gray values of area which are on both sides of the line are g1 and g2. This partial straight line edge becomes fuzzy because of defocus. The gray value of the point (a, b) which is near the edge is:

$$g(a,b) = g_1 \int_{-\infty}^{+\infty}\int_{-\infty}^{e(y)} G\left(\sqrt{(x-a)^2+(y-b)^2}, \sigma\right) \mathrm{d}x\mathrm{d}y + g_2 \int_{-\infty}^{+\infty}\int_{e(y)}^{+\infty} G\left(\sqrt{(x-a)^2+(y-b)^2}, \sigma\right) \mathrm{d}x\mathrm{d}y \tag{32}$$

Put the linear equation into this equation, and obtain an expression is:

$$g(a,b) = g_1 \Phi\left(\frac{d(a,b)}{\sigma}\right) + g_2 \Phi\left(\frac{-d(a,b)}{\sigma}\right) \tag{33}$$

The formula: d (a, b) from (a, b) to the straight edge of the vertical distance. Φ (·) is the standard normal distribution functions.
Standard normal distribution functions with:

$$\Phi(x) + \Phi(-x) = 1 \tag{34}$$

When (34) is put in (33), we get:

$$g(a,b) = g_1 \Phi\left(\frac{d(a,b)}{\sigma}\right) + g_2\left(1 - \Phi\left(\frac{d(a,b)}{\sigma}\right)\right)$$

$$g(a,b) = g_2 + (g_1 - g_2)\Phi\left(\frac{d(a,b)}{\sigma}\right) \tag{35}$$

According to the straight line distance from the point of the formula, we can get:

$$d(a,b) = \frac{| e(b) - a || e^{-1}(a) - b |}{\sqrt{[e(b) - a]^2 + [e^{-1}(a) - b]^2}} \quad (36)$$

e (b) and e-1 (a) were y = b and x = a is a point on straight edge, so to find out the exact edge position is very important to ensure accuracy.

From (35) we find that when known g1, g2, a point-to-edge distance d (a, b) and the extension function σ, can be determined point (a, b) the gray value g (a, b). Now we need to obtain σ and g1, g2.

Test the three point line around the area (ai , bi) (i = 1 ,2 ,3), and these three points to the straight line distance is different.

Depth correction

According to the edge of the defocusing case, we find the depth of the object point. Algorithm not only related to its accuracy, but also depends on the camera's optical system features. As the aberration, lens nonlinearity, optical system assembly error, will cause the results to large errors. As the object points in the coordinates x-y coordinates different, when the edge of the image plane coordinates are, even if the same σ value, the actual depth of the object will be different. Therefore calculated u by σ time, it should be added to position correction.

This algorithm can recover the depth by one image with defocus. This method reduces the calculation, saves storage resources and speeds up the operation efficiency, because it does not need calculate a few photos. However, this algorithm is restricted in some field, such as mobile phones and pocket cameras. Images taken with these devices do not have defocus area, because the size of sensors is too small. Since it is difficult to meet this function, the first algorithm is used to recover the depth information.

2.5 Rendering algorithm

2.5.1 Background

DOF is an important feature of imaging in human visual system. When we see the real-world, our eyes can automatically adjust the focus to adapt to different distance of objects. If we want to create movies and games which are more realistic, we should use effect of shallow DOF. Otherwise, the whole scene would seem true, but not natural.
In recent years, there were many on the depth of field rendering algorithm in the field of computer graphics. These algorithms mainly classified into the following three categories:

Post-processing filter

Potmesil and others [13,14] are representatives of post-processing filter and is also the first research scholars in DOF area. The algorithm uses the standard pinhole camera model for rendering scenes, and output value of each pixel of the depth z. According to the depth value z, aperture, focal length, they transform each point into circle of confusion with different size and different intensity distribution. Through covering all of its weighted average of circle of confusion the pixel can get the final value. Potmesil using Lommel intensity distributions function of the impact point of the surrounding pixels. Chen [15] also used a similar method to calculate the intensity distribution. Zhou et al [16] used the fuzzy filter to get blurred image, and then integrate with a clear image. This method can get real-time DOF.

Multiple rendering

The use of pinhole camera model, by changing the projection center of each fine and keep the same focus plane. Save the rendering result of the accumulation, and finally get an image with depth of field. However, many times the depth of field rendering[13] effects from heavy ghosting, lack of realism.

Reverse ray tracing

Kolb et al [10] geometric lens model Cook et al [11] distributed ray tracing using reverse tracking number to each pixel section of light. The color of each pixel gets these light colors values by weighted. The point is that the effect of superposition of

multiple beams of light on one image. This method avoids one to one mapping without depth of field defects, and it is different from the pinhole camera model.

2.5.2 Depth of Field Theory

Three-dimensional computer graphics model is a pinhole camera model used. Ideal pinhole camera hole is very small. The light emitted from a point source of light can enter only one hole. It is the point-to-point linear mapping relation. Hence, the ideal pinhole camera depth of field is infinity, in regard to any point within the domain in focus range.

DOF algorithm

In an image with depth of field, the pixel in the focus on the imaging surface is clear, and in the focus of two sides the pixel is blurred. Fuzzy degree and circle of confusion is directly proportional to the relationship. We integrate clear and fuzzy scenes to simulate DOF effect. Integration factor will be set normalized size of the circle of confusion.

Calculation of fuzzy information

In the virtual reality system (VR), the aperture, focal length and lens parameters are adjustable. In order to prevent the pixels in the front view translate to the back view, not only store the fuzzy degree of the points, but also store depth information. So that we can use multiple render targets, output the scene to a texture, and output the depth information and fuzzy information to another texture. In view of space, set the coordinates of object points as pos (x, y, z), set two plane a Znear and Zfar. Depth is z. Calculated circle of confusion diameter on Znear is Cnear. the diameter on the Zfar is Cfar. Comparing them to take the maximum as maxCoC.

$$maxCoC = max(Cnear , Cfar) \quad (37)$$

$$blurriness = | f \times z/(z - f) - f \times df /(df - f) | \times(D - f \times z/(z - f)) /max(Cnear , Cfar) \quad (38)$$

Image Pre-fuzzy

In the focus plane of the object point through the lens, it forms a circle of confusion on imaging surface. The screen is shown as a circular area consisting of multiple

pixels. It is the result of the interaction around the number of pixels. For better results, we used 2D separable Gaussian filter.

$$F = \left(\sum_{i=1}^{n} \sum_{j=1}^{n} P_{ij} \times C_{ij} \right) / S \tag{39}$$

F is the value of the filtered target pixel. Pij is the 2D Gaussian matrix of pixels. C is the pixel Pij corresponding Gauss coefficients. n is the dimensions of the matrix. S is a Gaussian matrix and all the coefficients.

Fused image

Now we can fuse these two images and get a DOF scene image:

$$color = a \times blurImage + (1 - a) \times originalImage \tag{40}$$

2.6 Previous work

In the area of camera's development, researcher never stop searching for a method for obtaining a better boken, such as changing the amount of aperture blade, increasing the number of the blade from 6 to more than 20, changing the structure of the lens, producing a special lens.

The following sections describe three different methods, on the impact on the depth of field.

2.6.1 Based on optical simulation

In the film camera, multiple exposure method is used to influence the depth of field and boken. This method works best, but because the film needs to use the camera fixed, any slight movement will cause the failure of the final image. Hence it is not popular.

Figure2.10: Comparison of STF the photo taken with 85mm, f/1.4 and f4 the top line (from left to right) The down line photo taken with 85mm, STF[2] This pictures are copied from Ref. [2].

There is a special mode called STF in a film camera named Alpha 7 that Minolta produced.

The Smooth Trans Focus (STF) mode enables almost all the Minolta AF lenses to produce a beautiful appearance of a defocused image. With the STF mode, the shape of defocused areas is faithfully reproduced. This mode sets the camera's film drive to multiple exposures, and stops down the aperture settings sequentially during the exposures to reduce the light transmittance in the lens periphery. In photography, bokeh is the aesthetic quality of the blur,[1] in out-of-focus areas of an image. The STF mode can make shape outside the focus smoother. The effect depicted in Figure2.10.

2.6.2 Using image processing software

When digital cameras became popular, all the sizes of digital compact camera sensor are far smaller than the film in film camera area. As a result, the lens focal length in the digital compact camera is not as the same as it is in film camera. The photo cannot capture the effect of shallow depth of field. To make more artistic photos, image post-processing software has become popular.

The following photo in Figure 2.11(bottom left) shows photo after image processing simulated the shallow depth of field by Photoshop. The right one is taken by real large aperture lens effect. This is the goal of depth of field simulation.

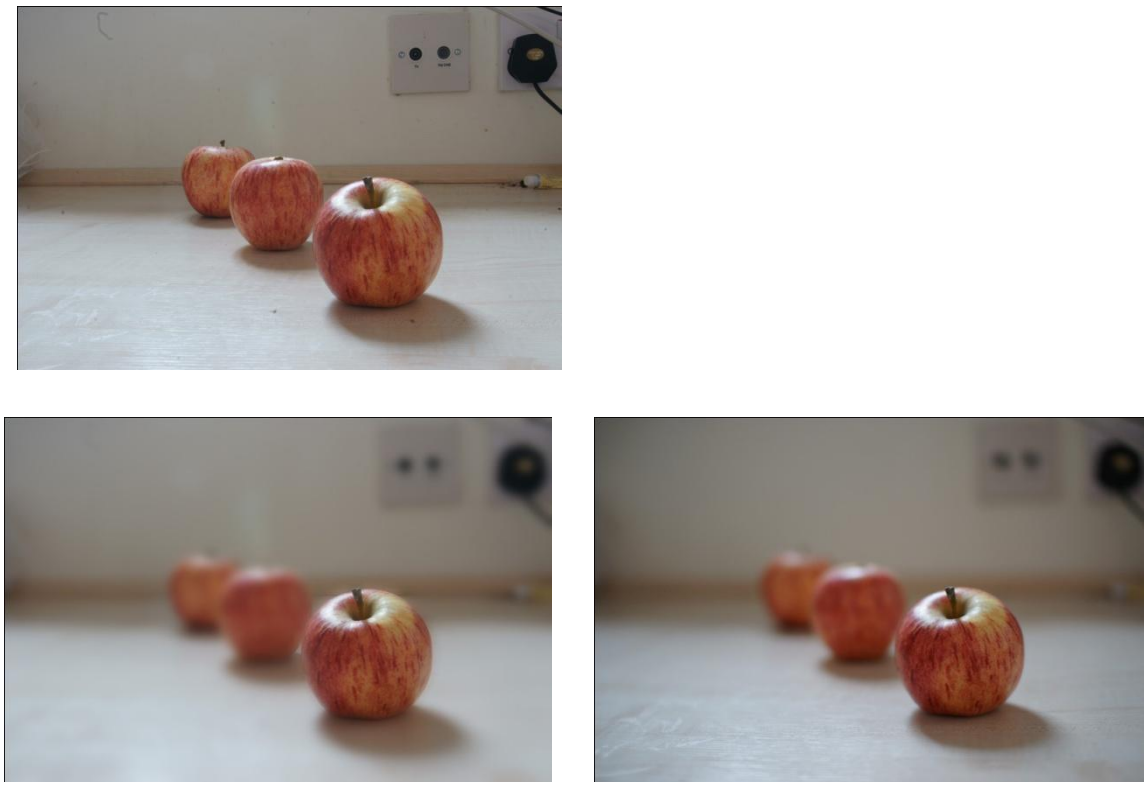

Figure 2.11:Comparison of original image with a large depth of field (top), after image processing by Photoshop to simulate slight depth of field (bottom left). The real lens takes an image with a large aperture (bottom right).

2.6.3 Pro Focus

Fuji has a similar function in its camera. The camera with this function was released in July 2009. The manufacturer claimed that it can take a shallow depth of filed effects as DSLR.

By using this mode the camera will combine 2 or 3 shots together. Through complex digital imaging technology, a crisp shot of the subject is created against a defocused background and foreground. The effect depicted in Figure 2.12. It is different from STF on Alpha 7, taken this 2 or 3 image with different focus plane, not different aperture value.

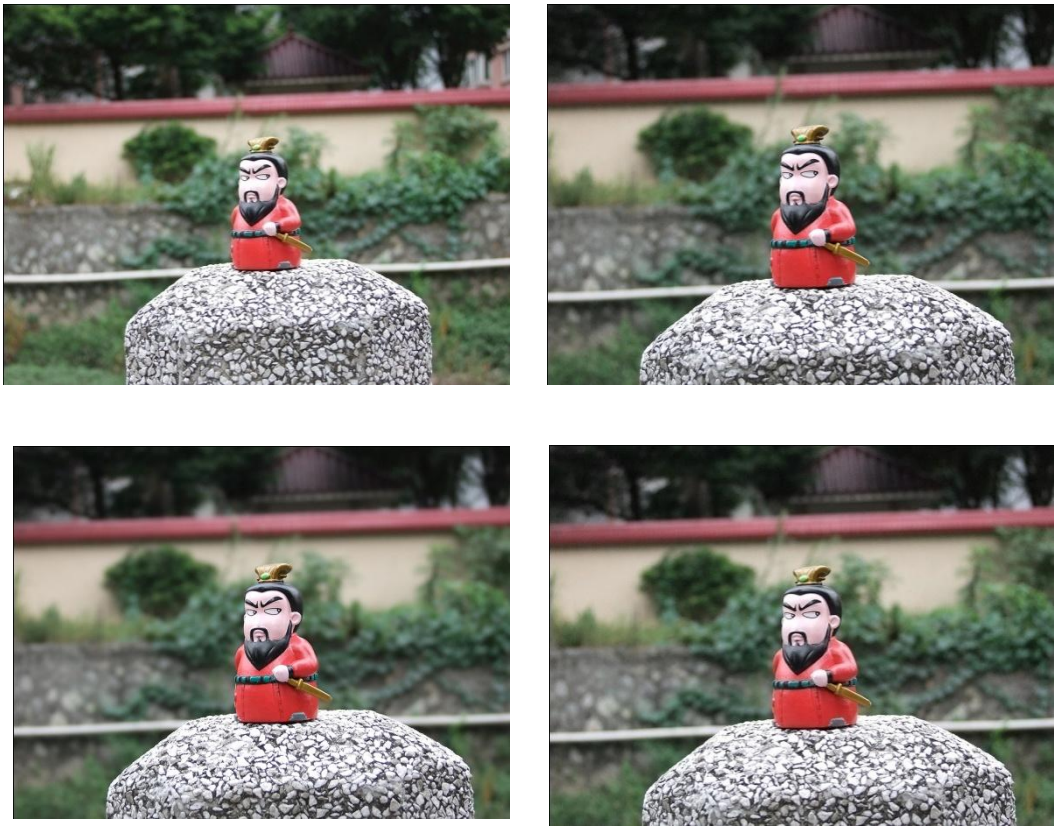

Figure 2.12: Comparison of pro focus with DOF photo by S205EXR in normal mode (top, left). Level 1 (top, left). Level 3 (bottom, left). Level 2(bottom, right) These pictures are copied from Ref. [4].

Although I did not know the algorithm of the function, but according to the details in the images I found out some useful information. In the combined image, there are some parts of the object edges blur as depicted in Figure 2.13. This shows that in combined processing, there is an algorithm to find the edge of the object. I found another detail in the images that have different noises in different areas, although these areas are with the same brightness and color as depicted in Figure 2.14. These shows that combined processing are performed with the use of image area selection and replacement.

Figure 2.13: Show some edge blur. Pro Focus closed (left) Pro Focus level 3(right) These pictures are copied from Ref. [3].

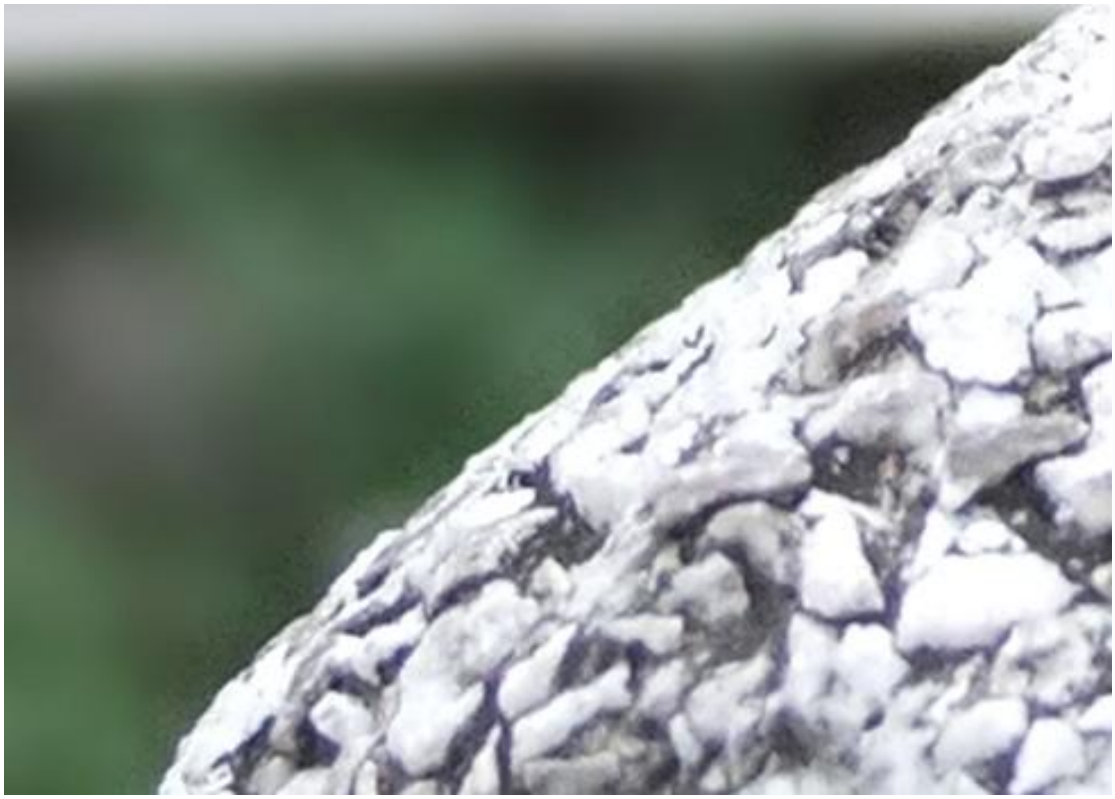

Figure 2.14: Shows the noise on the edge of the object. This picture is adapted from Ref. [3].

3 Algorithm and design

3.1 Overview

The algorithm used by the software, according to the areas of design can be divided into three parts: seeking depth, rendering and generating a new image, checking the aperture size and creating a vignette. The first part of the algorithm derives from the depth defocus and the second part of the algorithm is based on multiple rendering and ray tracing. The last part is based on summarizing some of the lens instructions from manufacturers, and through size of the aperture and focal length lens to simulate vignette. The effect of this program is close to the optical imaging results, so it needs to simulate all the characteristics of optical imaging.

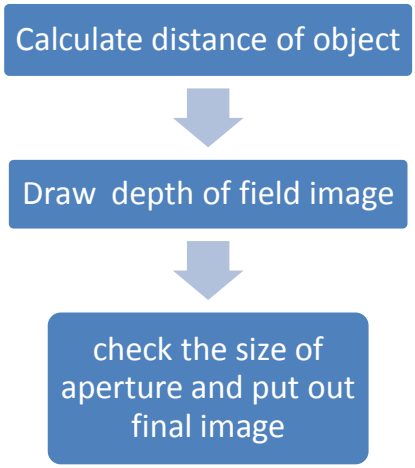

Figure 3.1: Whole program flow chart

3.2 Seeking depth algorithm and design

In chapter 2, this paper introduced a method for achieving depth from defocusing images according to a comparison of the two images which are taken with a different aperture. This part of the program algorithm and structure is shown below:

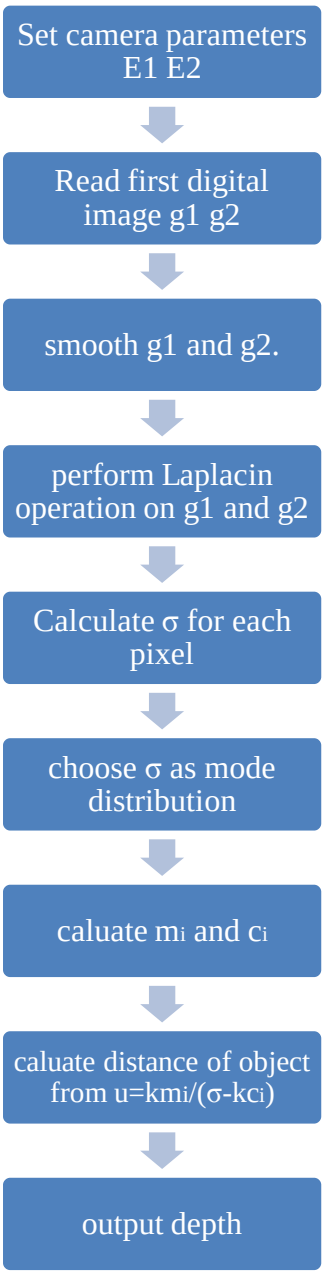

Figure 3.2: depth from defocus flow chart

First, two images are opened in JPEG format. They placed into a two-dimensional array, respectively. According to the algorithm needs, cvSmooth Function is used to filter each image. According to the derivation in chapter 2, depth information can be obtained from the following formula.

$$u = km_i / (\sigma_i - kc_i) \qquad [41]$$

mi and ci can be obtained.

$$m_i = -f_i s_2 / 2F_i \qquad [42]$$

$$c_i = -m_i(1/f_i - 1/s_i) \quad [43]$$

σ is fuzzy parameter, in the algorithm which is about the depth from defocus is a very necessary parameter and is used frequently.

$$\sigma_1 = a\sigma_2 + b \quad [44]$$

$$a = m_1/m_2\,,\ b = k(c_1 - ac_2) \quad [45]$$

In accordance with the formula, calculating each pixel and saving the result in the document in TXT format rendering an image algorithm and design

3.3 Rendering image algorithm and design

A two-dimensional array was used to store each pixel of depth information aone original photo was used as foundation, and rendered as a series of blurred images. Finally, this is blended between the original and pre-blurred image for better image quality.

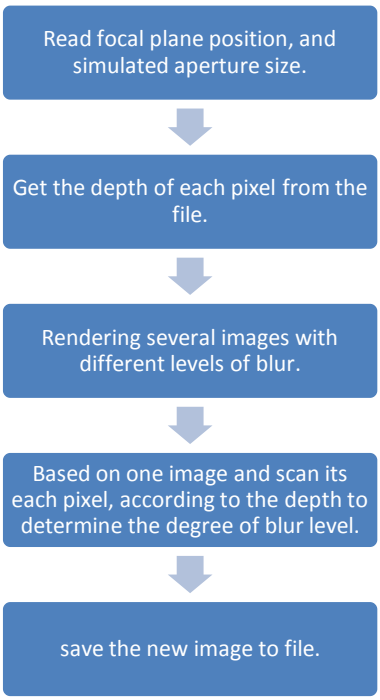

Figure 3.3: Multiple Rendering flow chart

This algorithm is used by companies and graphics companies in the field of real-time rendering. Such ATI is used and developed for this technique, because it does not require multiple render targets and is at better anti-aliasing.

However, this method produces a blurred outline around the edges of the objects. This is due to the blurred algorithm when the blur filter, impacts it surroundings. Because in real time rendering, it is always used for a dynamic output, maybe 60 or more frames every second, it is not very obvious. However, in photography, this phenomenon cannot be accepted. Although this algorithm is a very fast of achieving perfect photos, further research is needed into new algorithms.

Based on the ray tracing algorithm, and according to the features of the original image which will be used in rendering, I wrote a new algorithm. This method avoids the creation of fuzzy edges around the objects.

The first step is to calculate the distance between first point and a new focal plane. This will determine the diameter of its circle of confusion. According to the diameter of the circle of confusion, the first point depth can be compared with the surrounding point depths which are in the coverage area. If the first point depth value is larger than the comparison one, the first point will not affect the RGB value of the comparison point. Otherwise, in the formula in Chapter 2, new values of RGB can be calculated. The program does a loop in order to map each pixel, and at the end output the whole image.

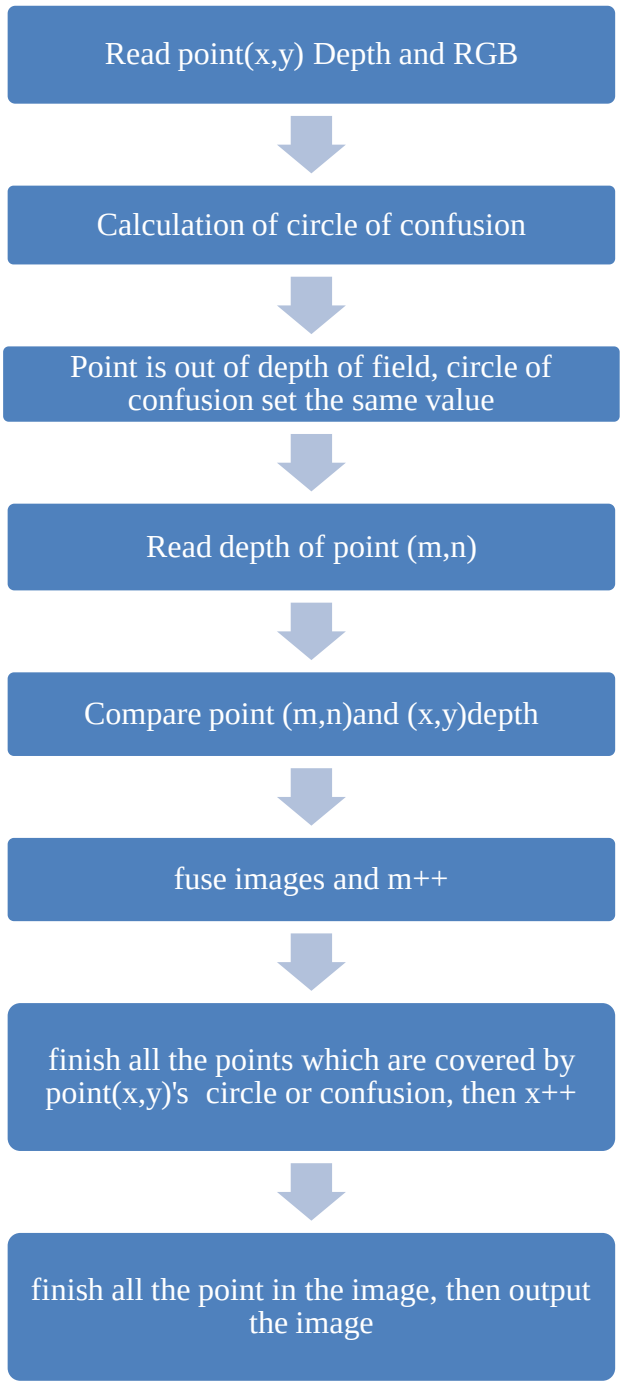

Figure 3.4: Precise Rendering flow chart

3.4 Create vignette algorithm and design

In photography and optics, vignetting is a reduction of an image's brightness or saturation at the periphery compared to the image center. This phenomenon is caused by the lens design, so different types of lens vignetting are not the same.

Different to each lens is the vignetting produced, so there is no certain formula. By observation we can understand the vignetting phenomenon produced only through a large aperture. Each pixel brightness value is affected by distance which is between it and the center of the image.

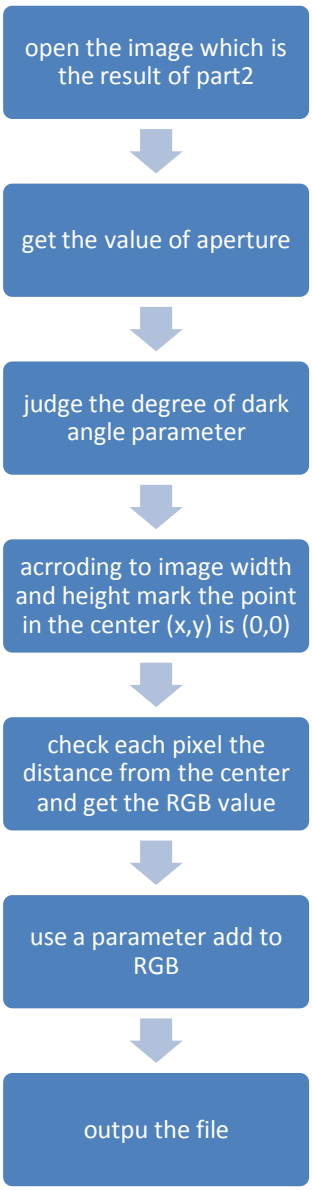

Figure 3.5: Precise Rendering flow chart

4 Experiments

4.1 Overview

This program consists of three parts. The first part is about the measurement depth. As the project does not require precise location, so focus only on the relative position of objects. The results of the depth information can be expressed by the color chart. The second part of the computation is based on the first part results. There are two kinds of rendering in the second part. One is multiple rendering, other one is based on ray tracing. In the following section, the final effect will be given, though the comparison some improvement can be found. The third part of the calculation is based on the results of the second part, gives the rendering photos and goal photos, and use the parameters of the application change the rendering results.

4.2 Results and Comparison

Images used in this program were taken by Sony A700, with lenses 18-55 / 3.5-5.6 and 30 / 1.4. With the information of depth obtained from the two images taken by 18-55lens with different aperture parameters, it is possible to simulate the effects of large aperture of F1.4 and F2.8.

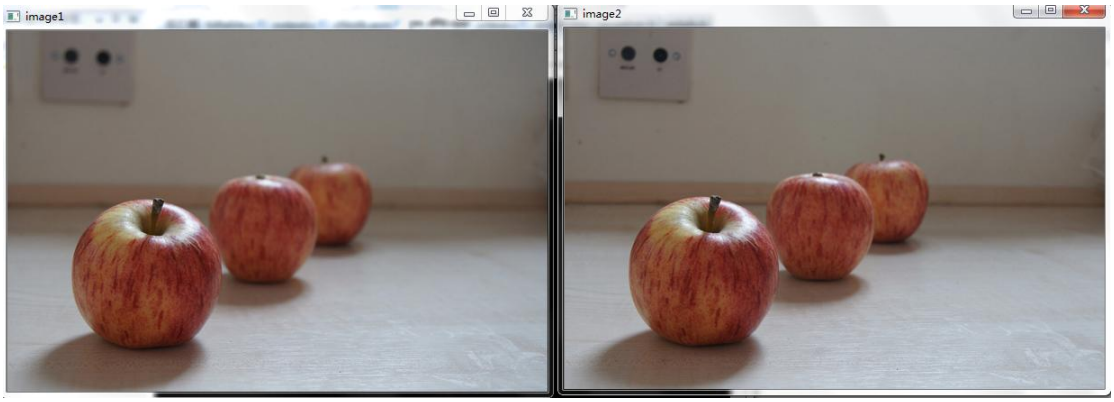

Figure 4.1 left is with aperture F4.5, right is with aperture F8

As figure shown, the basic two images are taken by F4.5 and F8. Using depth from defocus, the depth can obtain from this two images.

Multiple Rendering

Figure 4.2:show the pre-result in temporary storage

The algorithm is to repeatedly apply blur filter to the original image and save the processed images in temporary storage. Pictures shown above are pre-results which are firstly stored in the temporary storage. Secondly, according to the information of depth of each pixel, the virtual feature of each pixel (e.g. dot or arc) needs to be judged. In the final stage, processed images are merged and a new image is created.

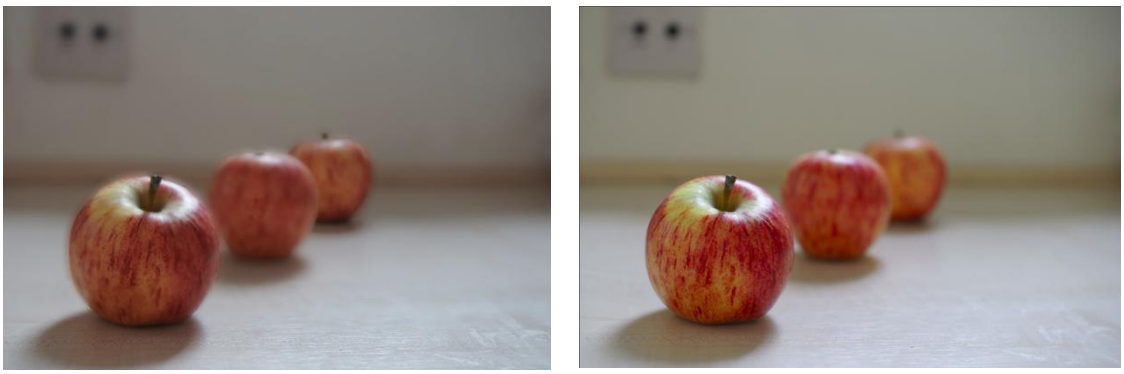

Figure 4.3: show the comparison of the result

The left image is simulation images by rendering, the aperture is F2.8.
The right image is taken by real camera with F2.8 aperture.

Simulation of the effects of shallow depth of field:

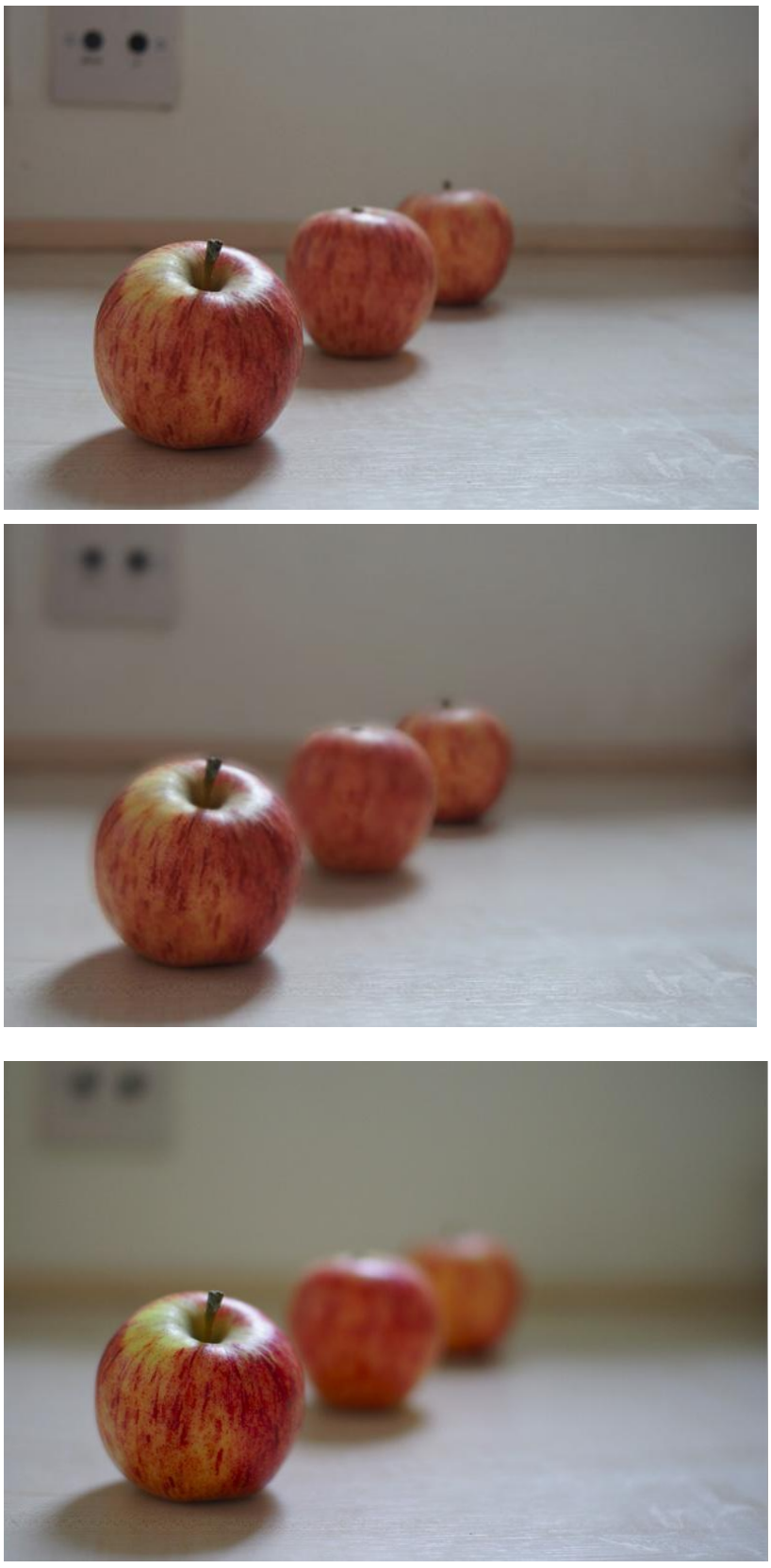

Figure 4.4:Top image is Original, Middle one is result of Multiple Rendering, Bottom one is taken by real lens F1.4

The results shown in Fig4.4 have basically achieved the effects of large aperture, despite of certain weakness, such as the colour leakage on the edge of objects. The reasons of such phenomenon lie in the employment of fuzzy filter. Due to inaccurate calculation of parameters, the degrees of blur between the middle and the farthest apples are identical, which can be improved by amending the parameters.

For static picture output, each picture is presented in front of the audience for a long period, which is different from video in which each frame is displayed for only a few seconds. Hence, rigid requirements are imposed into static images. Problems like colour leakage are not allowable. To solve the problem of colour leakage, another algorithm of precise rendering is proposed as follows.

Precise Rendering

The algrithm is performed by taking calculation based on point by point scanning. The shape of each pixel can be obtained through its depth value and the location of the focal plane. If the measured pixel located on the focal plane, it will generate a clear virtual image. If, on the other hand, the pixel is out of the focal plane but within the depth of field, the size of the confusion circle can be obtained by extracting the information of depth and focal plane. When the object is out of the depth of field, its confusion circle is set to be a fixed value, which does not enlarge or shrink as the focal plane changes.

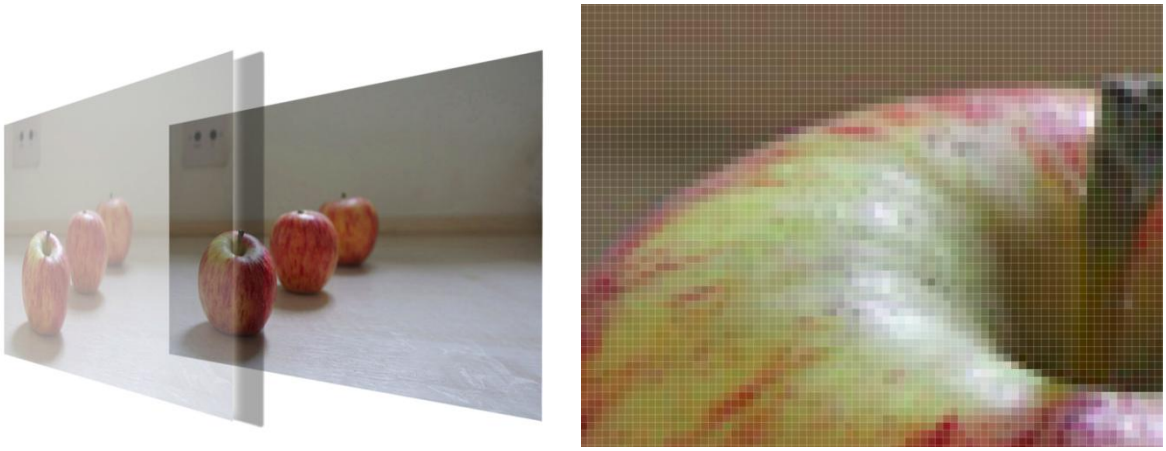

Figure 4.5: show the two space store images, and scan and calculate point by point

The first step is to calculate the size of each confusion circle. Before calculating the size of CoC for each pixel within the confusion circle, it is required to judge the location relationship between the goal pixel and any other point within the confusion circle. If the opponent point locates closer to the camera than the goal pixel does, the colour of the goal pixel will not be merged with the opponent point. Otherwise, the colour value of the goal pixel needs to be merged with that point. Each pixel will be compared with all the other points in the confusion circle.

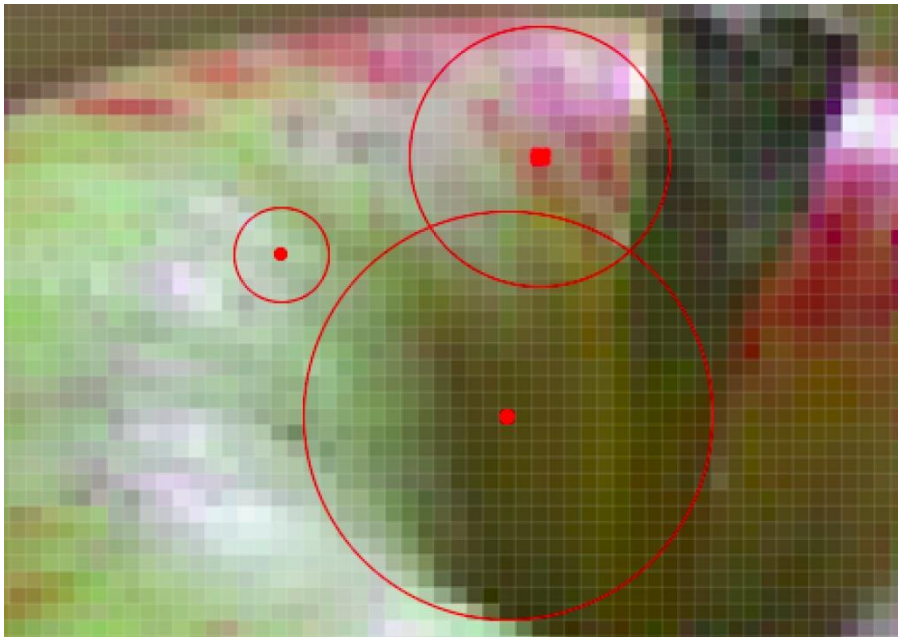

Figure 4.6: give the example ,the circle of confusion size is different.

Figure 4.7: the result of Precise Rendering

Although the amount of computation is significant, this algorithm is able to render an image with shallow depth of field from an image with depth of field, without causing colour leakage. According to the results of the figures shown below, despite of certain problems with the background due to the precision of calculations, it managed to solve the problem of colour leakage after fuzzy the objects.

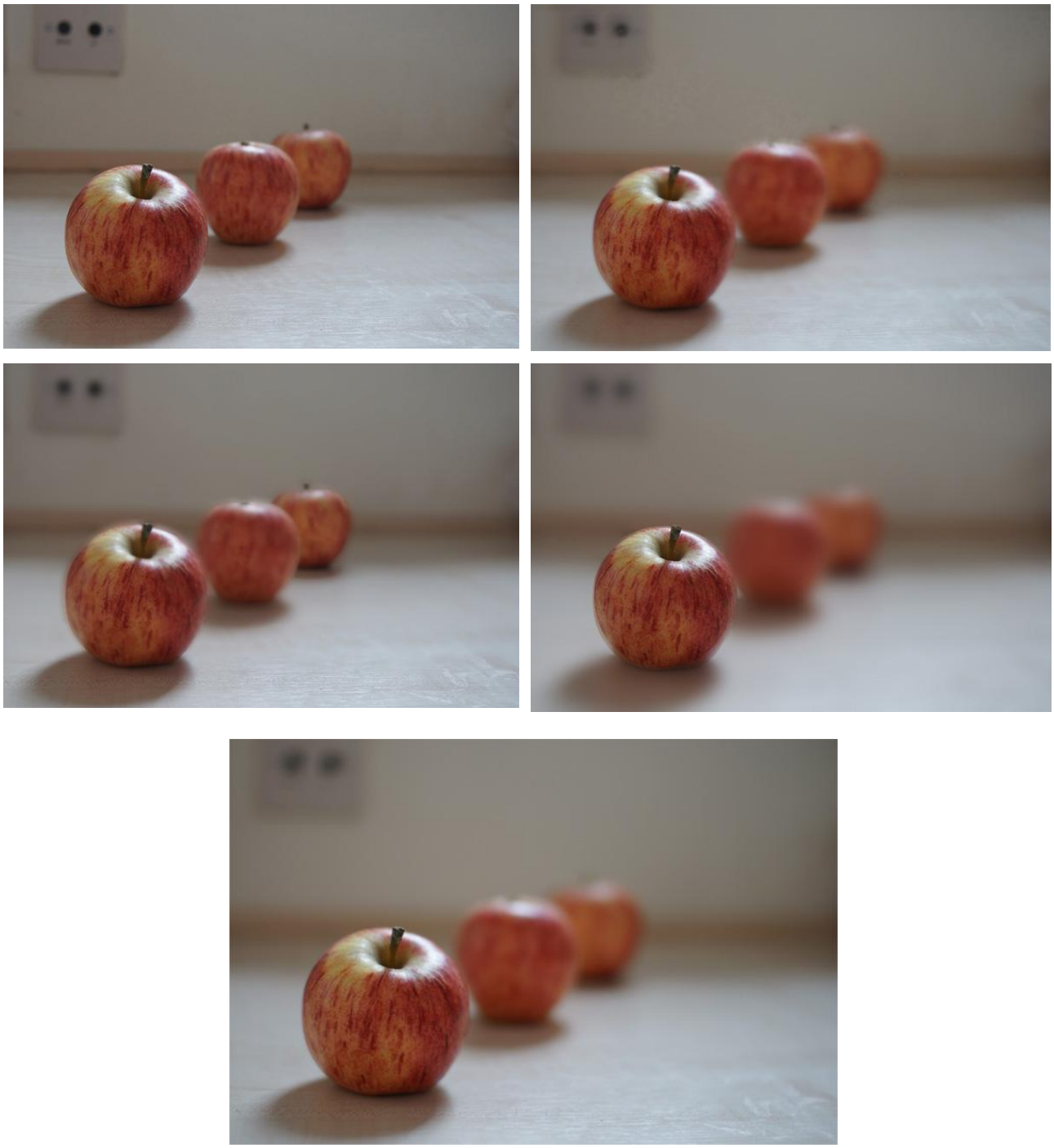

Figure 4.8: Comparison different results with original image and image by real-lens taken

The upper left image is the original image, taken with F4 aperture. All the others are generated from the original one by using the technique of rendering. Because the aperture is not large enough, the depth of field is high, hence all objects in the picture are clear. As a result, the image appears messy, without focusing on the main object.

The upper right image with shallow depth of field is derived by Precise Rendering. One of the advantages of this image is that the problem of colour leakage is totally

solved. Besides, it displays the gradual change of the darkness of desk from clarity to fuzzy. There are some errors in the background due to the immaturity of algorithm.

The middle left image is derived by multiple rendering. The gradual change of the darkness of the desk surface is displayed well but the colour leakage on the edge of the object is obvious.

The middle right image is processed by Photoshop. Blur filter is also used. Because of the employment of the mask and layer processing techniques, the output colour leakage is not obvious. However, there is not apparent gradual change from clarity to fuzzy. Although the relationship of location between the objects in this image is simple, the depth of field can be changed by using the Photoshop image processing. If a complex scene is taken into account, it is difficult to do so.

The image at the bottom, taken by large aperture lens with the size F 1.4, serves as a reference image used for comparison.

Vignetting Phenomenon

Through comparison of results generated above, it can be found out that except for the shallow depth of field and good out-of-focus imaging, lens with large aperture can also produce vignetting effects. This is the characteristic of all lens with large aperture, particularly wide-angle lens. Therefore, this program adopted the function of simulating vignetting effects according to the size of aperture.

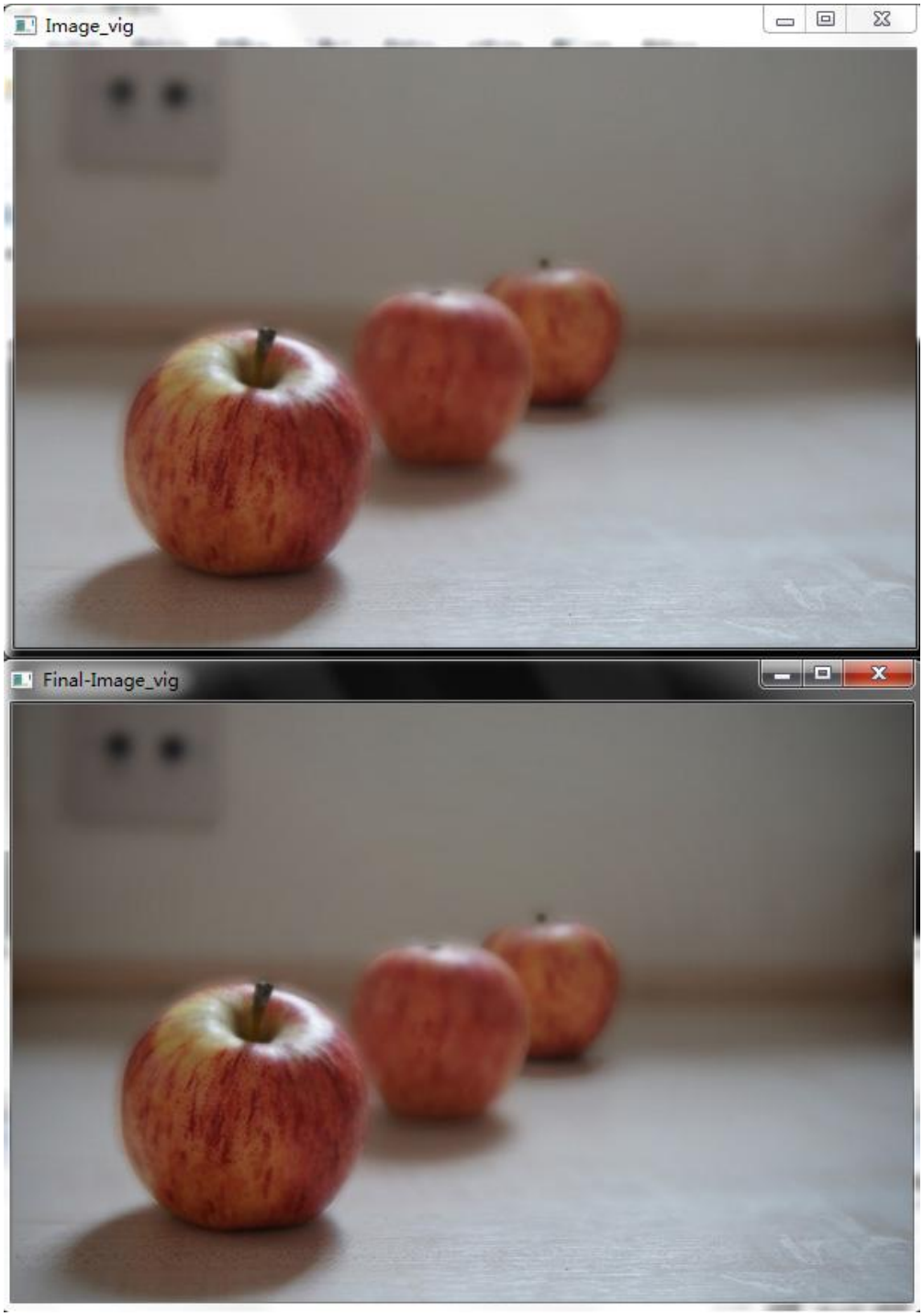

Figure 4.9: The effect of vignettie

The upper image is derived from the previous part by multiple rendering. Based on the image above, the image below shows the vignetting effects with F1.4 aperture. Through the comparison with the goal image taken by real lens with F1.4 aperture, it may arrive at a conclusion that by adding vignetting effects, the transition of brightness and darkness is relatively natural, making photos more artistic.

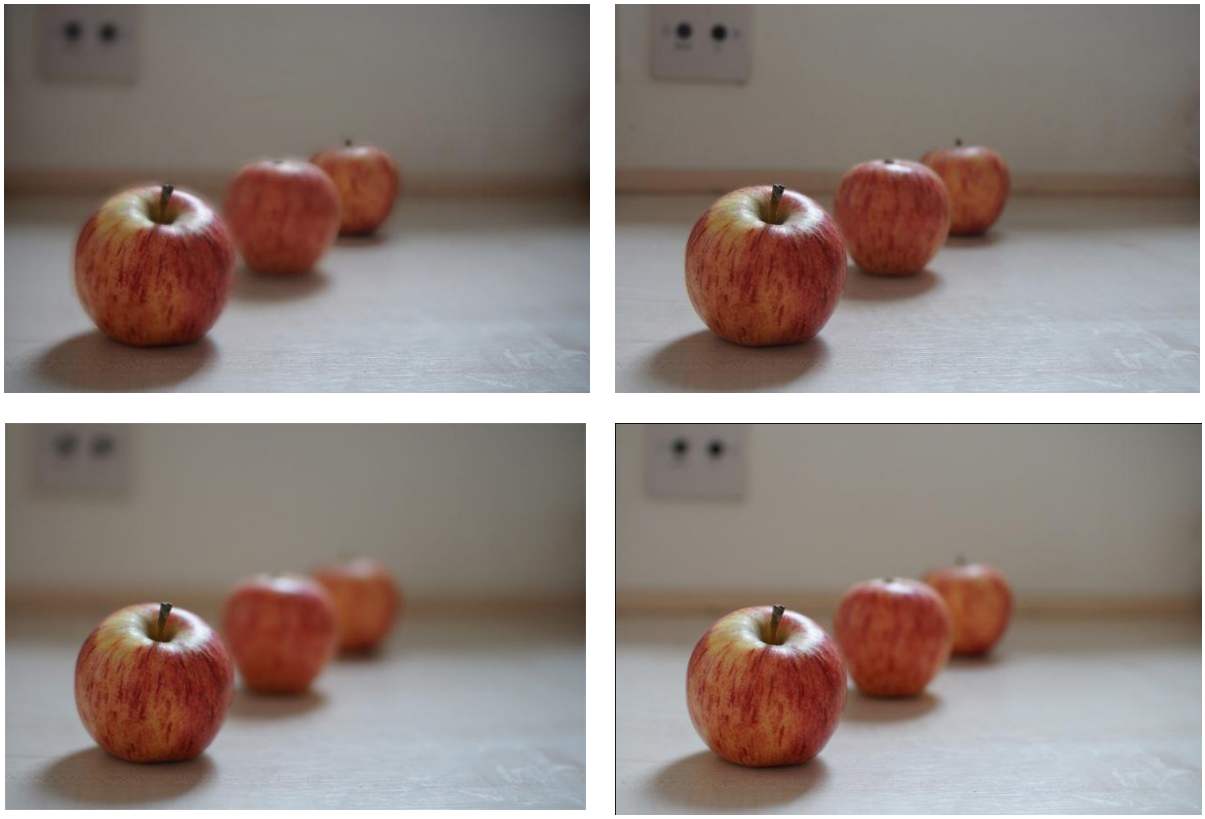

Figure 4.10: Comparison the results with goal images

The upper left image is the effect figure simulating the aperture F1.4 by multiple rendering. The upper right image is the effect figure simulating the aperture F2.8 by multiple rendering. The lower left image is the real photo taken by camera with aperture F1.4. The lower right image is the real photo taken by camera with aperture F2.8.

4.3 Simulation of phenomena

Two photos which are taken with different defocus can obtain the depth information, and through another photo with large depth of field, this program can render an image which is the focal plane in anywhere in the scene.

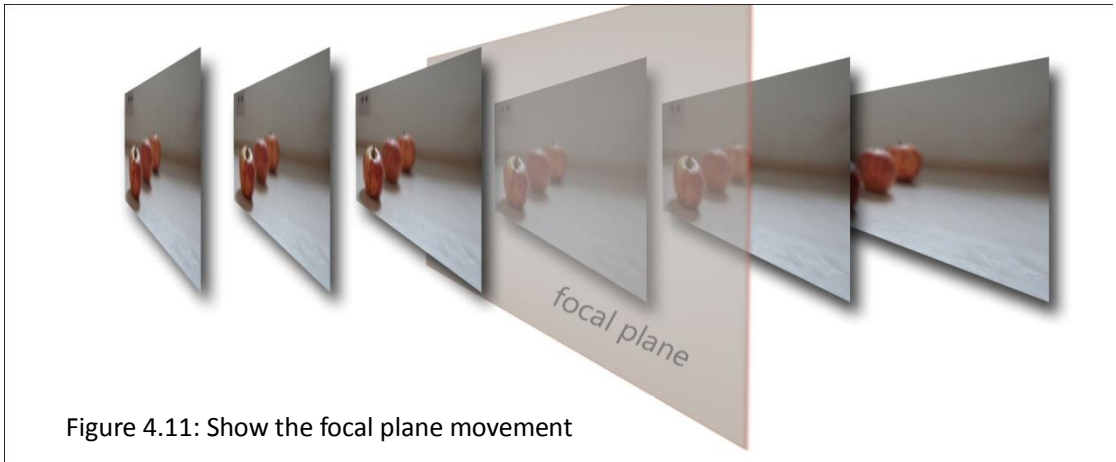

Figure 4.11: Show the focal plane movement

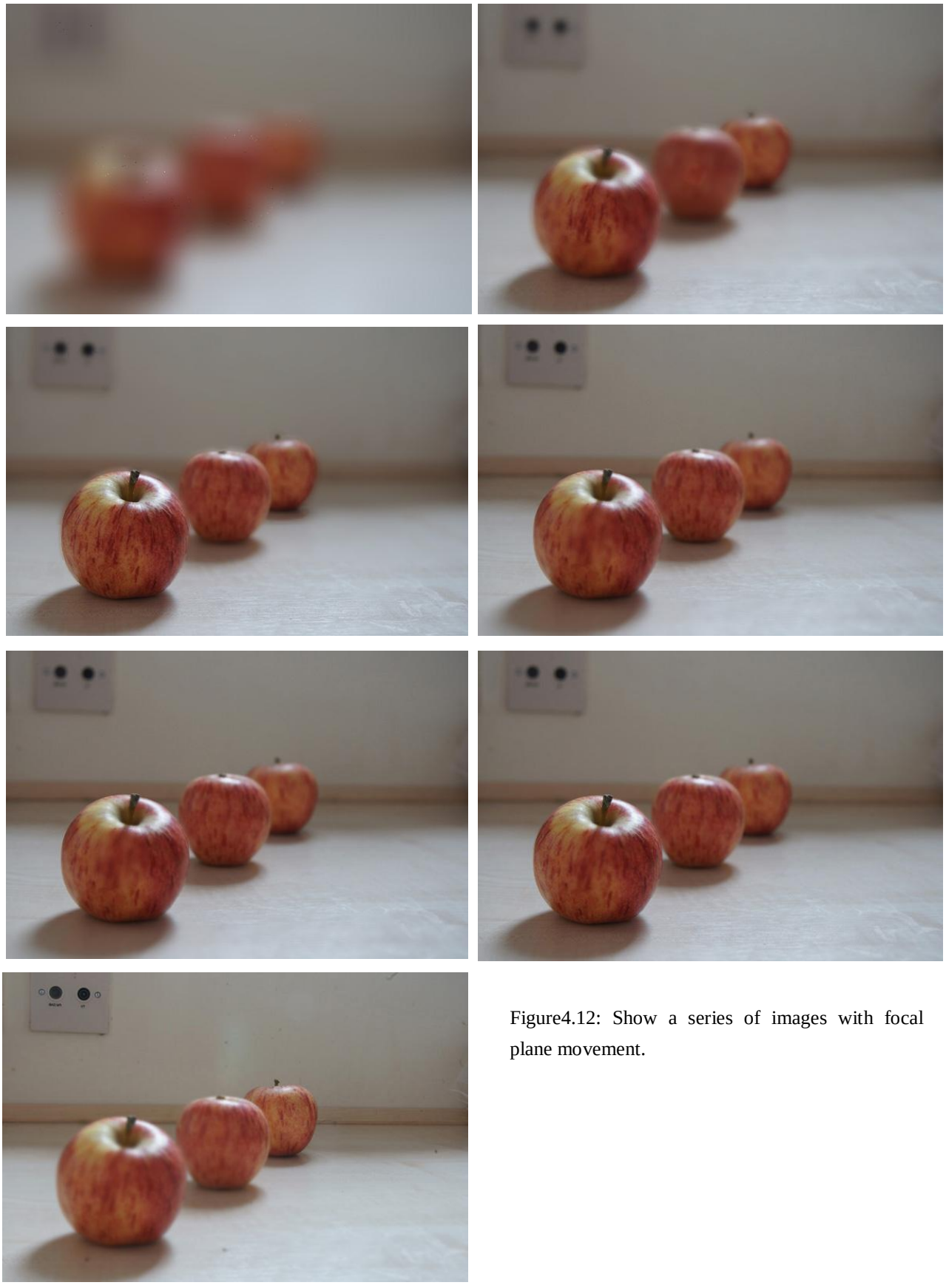

Figure4.12: Show a series of images with focal plane movement.

The series of photos shown above display the effects of changing the location of focal plane. From left to right and top to bottom, the focal plane is placed further from the

camera.

5 Conclusion

In this project, the main aim is to use image processing method to produce shallow depth of field. By comparison and analysis, I selected depth from defocus algorithm, and combined it with the rendering algorithm to create a new image. Finally, in order to make the results more close to the real image taken by large aperture, I add vignette effect in to the final image.

From the invention of the camera to the present, there are times that technologies focusing on optimizing bokeh, were born. Bokeh is the only way to describe the depth of field in a photograph. It is mentioned in the chapter 2, the early camera manufacturers increased the lens aperture blades and changed shape of lens leaves to get higher quality bokeh. In the 90's, the camera body was incorporated with program exposure to change the out-of-focus areas of an image. Digital program control plays a more and more important role in modern cameras; however, the final photo was only a result of optical principles. Up until around 2005, with the popularity of digital cameras, photography has reached a popularity level it never had. As designed for cost reasons, digital cameras, unlike film cameras, have limited ability to capture shallow depth of field. So, people think about using image-processing software such as Photoshop for post-process and simulation of a shallow depth of field. The combination of digital post-process and optical principles is beginning to take roll in final effect with the increase in computing power. In late 2009, camera-integrated function of simulation of shallow depth of field made its first debut. Although the result is not perfect, but it indicated that image processing technology is being applied to the simulation of the depth of field.

From the invention of the camera to the present, each period, some technologies which change the depth of field were born. Bokeh is the only way to describe the depth of field in a photograph. This has been mentioned in Chapter 2, the early camera manufacturers increase the lens aperture blades and changed the shape of the lens leaves to achieve a higher quality bokeh. In the 1990's, the camera body joined the program exposure to change the image to out of focus. This was joined by the digital program factor. However, the final photo is only in accordance with optical imaging law. Around 2005, with the popularity of digital cameras, photography has never before reached a higher popularity level. As digital cameras are designed for cost reasons, they are not like film cameras which have a very shallow depth of field. Therefore, people think using image processing software for post-processing and simulating depth of field, such as using Photoshop can be effective. The final effect of the photo was based on a combination of digital computing and optical laws. With the increase in computing power, in late 2009, there was a new integrated camera function which simulated slight depth of field. Although the result is not perfect, it has indicated that the image processing technology is applied to the simulation of the depth of field.

Algorithms in this project move common technologies into the photography arena. One is used in the medical field and computer vision, another one is widely used in games and movies.

We also need to take into account the characteristics of photographic equipment, before the final image effect can be simulated. This is a comprehensive and interdisciplinary project. The project originated through one of my ideas and through the guidance of my supervisor. I research a variety of algorithms about shape from defocus, depth from focus, depth from defocus. Through selection and comparison, I chose depth from defocus methods and chose the algorithm to obtain depth from the two photographs taken by different apertures. As the algorithm is based on two images with different aperture defocus, it obviates some problems such as object size change in the use of defocusing by changing the focal plane. Therefore, it can reduce a number of unnecessary calculations. It does not require standardization calculation about the object size. It improves operational efficiency and reduces the computing time. In the rendering part, I offered two alternative calculation methods: the first a quick calculation based on multiple rendering, the second method from my idea and tests, which has some similarity with ray tracing. In this calculation, I scan each pixel as an object. By comparing the depth of each pixel, this rendering method can solve the problem of blurred edges around sharp objects. In the last part I conducted research about the features of camera lenses in photography, and according to the observation and analysis of a large number of pictures taken by myself, I obtained the rules for vignetting. After finishing the code for the algorithm, I compared the test results.

The advantages of this project:

- By calculating based on a photo to render a new one, noise was at the same level in different areas.

- Obtained defocus image by changing the aperture, and thus can avoid objects' size changing as in defocusing an image by changing the focal plane. It does not need to standardize the size of the calculation, so it reduces running time.

- Multiple rendering to reduce computation time which does not need to render multiple targets. Improved ray tracing algorithms for accurate results can effectively solve the phenomenon with obvious edges aroundthe clear objects caused by the fuzzy filter.

- Through the additional vignetting effect the final photo can be produced in the same way as one which has been taken by real large aperture, and with a realistic shallow depth of field effect.

Several shortcomings:

- As an object is taken with a different aperture, so the accuracy depends on the selection of aperture size. In the rendering part, because it need store multiple images, it need open large memory space to store. And it used the fuzzy algorithm. There will be edge around the clear object. Using the changed ray tracing algorithm can solve the edges blot problem although each point and coverage by its circle of confusion needs to be calculated so each pixel can be recounted several times. For a high resolution image, the calculation is the large size and because rendering a new image is based on a photo with the depth of field, some points in the out of focus area will be calculated the same as points in focus. According to the table and formula in the chapter 2, we can know that if an object is located beyond a near or far boundary, the defocus level of the object is the same. Therefore, some calculations are in fact useless.

I hope that this algorithm can be applied in a camera. When combined with an auto focusing system, it can gather more useful information from the camera parameters. In the current stage, there are many areas for improvement, such as the storage structure and function which can be changed, as well as increasing the image alignment algorithms. Since I want to use the program in a camera, the case of high-ISO noise needs to be better understood. In future work I need to conduct research into high ISO image quality, including high noise filtering, and high noise impact on measurement distance.

6 FUTURE WORK

There are many parts of this project can be improved and expanded. According to the application platform, different functions could be extended. If this project is used as image-processing software, as the working platform has a strong computing power, so the existing algorithms in the second part could be improved. It can be changed to using real-time rendering depth of field algorithms. And using OpenGL can make the program have interaction.

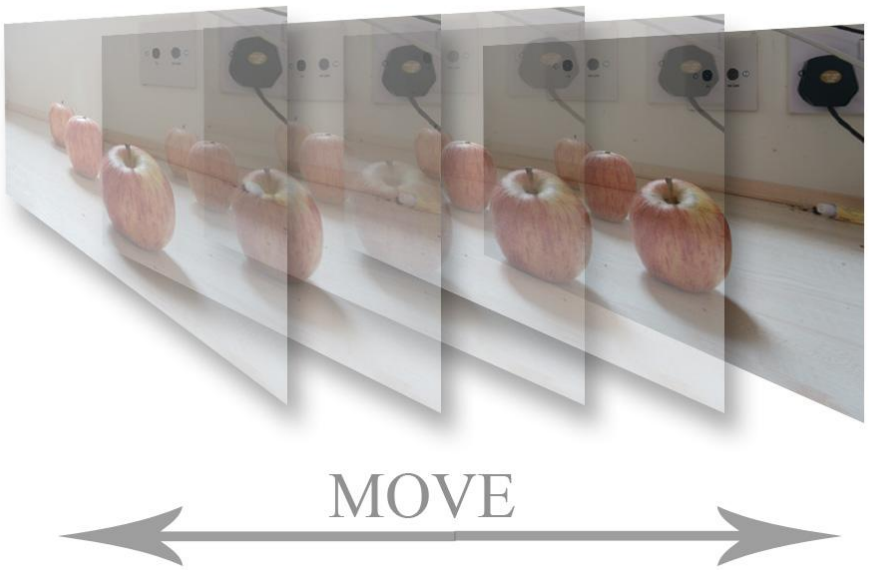

Figure: 6.1: Show the effect of real-time rendering

The use of real-time rendering algorithms can add a function to move the location of focal plane and to change the size of aperture to the application. With the information of an extreme depth of field, a photo taken at a small aperture, which is similar as taken with a pinhole camera, could be used to render an image with very shallow depth of field. Therefore to create an effect this was only achievable with wide-open aperture not long ago. This technology can take users back to shooting time and scene. Because users are no longer facing the two-dimensional images, a three-dimensional scene can be rebuilt. According to the change of parameters of virtual cameras, the result could be viewed directly on the screen. Then users can save rendered images into file. This image processing software will provide great convenience to users.

Similarly, the algorithm in the third part of the additional vignetting effect could also be controlled by the value of parameters inputted by keyboard; the real-time effect can be rendered and saved to users' needs.

If this algorithm is integrated as a function of the camera, the second part of the algorithm for optimization is very necessary. As most cameras reach ten million pixels level, it is almost impossible for cells inside cameras to render a new image with shallow depth of field point by point. Storage and computing capacity of all current cameras' hardware are not powerful enough. So images can be reduced to one tenth or one-twentieth of the original size for calculation. When the application do rendering in the second part, value of depth of field could be used for an area containing several pixels, instead of one pixel. Because the program runs inside the camera, the camera lens can get the focal length and aperture information, and also can get the focal plane distance information from auto focusing system. By the depth of field formula in chapter 2, depth of field can be calculated between the near boundary and far boundary. There are two photos, the first one is taken by large aperture with slight depth of field, and the second is taken by small aperture with large depth of field. In the calculation of depth, points beyond boundary are marked and then are not recalculated in rendering period. This can reduce computation and improve efficiency.

Bibliography

[1].Gerry Kopelow (1998). How to photograph buildings and interiors (2nd ed.). Princeton Architectural Press. p. 118–119. ISBN 9781568980973.

[2].Image from website:
http://www3.xitek.com/testreport/xitek/m135stf/oof-prin.htm

[3].Images from website:
http://www.dyxum.com/dforum/maxxum-7-stf-mode-experiments_topic26328_page1.html

[4]. Images from website:
http://www.dcfever.com/news/readnews.php?id=3172
http://dcbbs.zol.com.cn/61/752_605630.html

[5] PENTLAND A P. A new sense for depth of field[J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 1987, 9 (7) : 5232531.

[6] SUBBARAOM. Parallel depth recovery by changing camera parameters[C] Proc of the 12nd International Conference on Computer Vision. Floride: IEEE Computer Society, 1998: 1492155.

[7] SUBBARAOM,SURYAG. Depth from defocus: a spatial domain approach[J]. International Journal of Computer Vision, 1994, 13(3) : 2712294

[8] VINAYA, NAMBOOD IR I P, CHAUDHUR I S. On defocus, diffusion and depth estimation[J]. Pattern Recognition Letters, 2007,28 (3) : 3112319.

[9] ZIOU D, DESCHENES F. Depth from defocus estimation in spatial domain[J]. Computer Vision and Image Understanding, 2001,81 (2) : 1432165.

[10] FAVAROP, SOATTOS. Shape and radiance estimation from the information divergence of blurred images[C] Proc of IEEE Computer Vision and Pattern Recognition. London: Sp ringer2Nerlag, 2000:7552768.

[11] PARKSY. An image-based calibration technique of spatial domain depth-from-defocus[J]. Pattern Recognition Letters, 2006, 27(12) : 131821324.

[12] MEER P, WEISS I. Smoothed differentiation filters for images[J].Journal of Visual Communication and Image Rep re sentation,1992, 3 (1) : 58272.

[13] ULDEJD. LIERER van. Fast perception-based depth of field rendering[C] Proc

of ACMVRST. Seoul: [s. n.] , 2000: 129-133.

[14] POTMESIL M, CHAKRAVARTY I. A lens and aperture camera model for synthetic image generation [J] . Computer Graphics,1981, 1 5(3) : 297 -305.

[15] POTMESIL M, CHAKRAVARTY I. Synthetic image generation with a lens and aperture camera model[J] . ACM Trans o n Graphics,1982, 1 (2) : 85-108 .

[16] CHEN Y C. Lens effect on synthetic image generation based on light particle theory[J] . The Visual Computer, 1987, 3 (3) : 125-136.

[17] Park J S. Interactive 3D reconstruction from multiple images:a primitive-based approach [J]. Pattern Recognition Letters, 2005, 26(16) : 2558-2571

[18] Intel Corporation. Open Source Computer Vision Library Reference Manual [S]. 2001-12

[19] Pollefeys M, Koch R, Van Gool L. Self-calibration and Metric Reconstruction in Spite of Varying and Unknown Internal Camera Parameters[C]. Proc. of International Conference on Computer Vision, Bombay, India, 1998: 90

[20] Hartley R I, Zisserman A. Multiple View Geometry in Computer Vision [M]. Cambridge University Press, 2000

[21] Wu Fuchao, Li Hua, Hu Zhanyi. A New Camera Self-calibration Method Based on Active Vision System [J]. Chinese Journal of Computers, 2000，23(11): 1130-1139

[22] XIONG Y, SHAFER SA. Depth from focusing and defocusing[C] Procof DARPA Image Understanding Workshop. New York: IEEE,1993: 6182673.

Appendices: Source code

```
#include "stdafx.h"
#include "highgui.h"
#include <cv.h>
#include <highgui.h>
#include <stdio.h>
#include <iostream>
#include <string>
#include <iomanip>
#include <fstream>
#include  <math.h>



using namespace std;

int _tmain(int argc, _TCHAR* argv[])
{


     const char* imagename1 = "4b.jpg";


     cv::Mat img1 = cv::imread(imagename1);    if(img1.empty())
    {
        fprintf(stderr, "Can not load image %s\n", imagename1);
        return -1;
    }

    if( !img1.data )        return -1;

     cv::namedWindow("image1", CV_WINDOW_AUTOSIZE);
     cv::imshow("image1", img1);

     const char* imagename2 = "8b.jpg";


     cv::Mat img2 = cv::imread(imagename2);    if(img2.empty())
    {
        fprintf(stderr, "Can not load image %s\n", imagename2);
        return -1;
    }
```

```
if( !img2.data )        return -1;

 cv::namedWindow("image2", CV_WINDOW_AUTOSIZE);
 cv::imshow("image2", img2);
 cv::waitKey();


float imgdepth[400][600];
float aperture;
 float focalplane;
 IplImage *img=cvLoadImage("4b.jpg",1);
IplImage *imgfinal=cvLoadImage("4b.jpg",1);

fstream infile;
infile.open("2.txt");
// string temp;
float t1;
 for(int i=0;i<400;i++)
 {
    for(int j=0;j<600;j++)
        {
            infile>>t1;

            imgdepth[i][j]=t1;

        }

}
 infile.close();
  fstream infile2;
 float t2;
  infile2.open("aperture.txt");
  infile2>>t2;
 aperture=t2;
  infile2.close();


   fstream infile3;
 float t3;
  infile3.open("focalplane.txt");
  infile3>>t3;
 focalplane=t3;
  infile2.close();
```

```
 printf("focalplane=%f",focalplane);

CvScalar s1,s2,s3,s4,s5,sfinal,tempimg;

 cvSmooth(imgfinal,imgfinal,CV_BLUR,3,3,0,0);
 IplImage *dst1 = NULL , *dst2 = NULL, *dst3 = NULL , *dst4 = NULL, *dstemp = NULL;


 dst1 = cvCreateImage(cvSize(img->width,img->height),IPL_DEPTH_8U,3);
 dst2 = cvCreateImage(cvSize(img->width,img->height),IPL_DEPTH_8U,3);
dst3 = cvCreateImage(cvSize(img->width,img->height),IPL_DEPTH_8U,3);
 dst4 = cvCreateImage(cvSize(img->width,img->height),IPL_DEPTH_8U,3);
 dstemp= cvCreateImage(cvSize(img->width,img->height),IPL_DEPTH_8U,3);

 cvSmooth(imgfinal,dst1,CV_BLUR,3,3,0,0);
 cvSmooth(imgfinal,dst2,CV_BLUR,9,9,0,0);
cvSmooth(imgfinal,dst3,CV_BLUR,15,15,0,0);
 cvSmooth(dst3,dst4,CV_BLUR,50,50,0,0);
 cvSmooth(dst4,dst4,CV_BLUR,3,3,0,0);


 for(int i=0;i<img->height;i++)
 {
    for(int j=0;j<img->width;j++)
      {
    s1=cvGet2D(img,i,j); // get the (i,j) pixel value
      s2=cvGet2D(dst1,i,j);
      s3=cvGet2D(dst2,i,j);
      s4=cvGet2D(dst3,i,j);
      s5=cvGet2D(dst4,i,j);
      tempimg=cvGet2D(dstemp,i,j);

      if(aperture<2.0)
      {
        if( fabs(imgdepth[i][j]-focalplane)<20.0)
        {    sfinal.val[0]=s1.val[0];
            sfinal.val[1]=s1.val[1];
            sfinal.val[2]=s1.val[2];
        }
   if( fabs(imgdepth[i][j]-focalplane)>=20.0&&fabs(imgdepth[i][j]-focalplane)<30.0)
        {    sfinal.val[0]=s2.val[0]*0.7+0.3*s1.val[0];
            sfinal.val[1]=s2.val[1]*0.7+0.3*s1.val[1];
            sfinal.val[2]=s2.val[2]*0.7+0.3*s1.val[2];
```

```
    }

if( fabs(imgdepth[i][j]-focalplane)>=30.0&&fabs(imgdepth[i][j]-focalplane)<40.0)
    {    sfinal.val[0]=s3.val[0]*0.8+0.2*s1.val[0];
        sfinal.val[1]=s3.val[1]*0.8+0.2*s1.val[1];
        sfinal.val[2]=s3.val[2]*0.8+0.2*s1.val[2];
    }

if( fabs(imgdepth[i][j]-focalplane)>=40.0&&fabs(imgdepth[i][j]-focalplane)<80.0)
    {    sfinal.val[0]=s4.val[0]*0.9+0.1*s1.val[0];
        sfinal.val[1]=s4.val[1]*0.9+0.1*s1.val[0];
        sfinal.val[2]=s4.val[2]*0.9+0.1*s1.val[0];
    }

if( fabs(imgdepth[i][j]-focalplane)>=80.0)
    {    sfinal.val[0]=s5.val[0];
        sfinal.val[1]=s5.val[1];
        sfinal.val[2]=s5.val[2];
    }

         tempimg.val[0]=int(imgdepth[i][j])%255;
        tempimg.val[1]=int(imgdepth[i][j])%255;
        tempimg.val[2]=int(imgdepth[i][j])%255;

  }

  else if(aperture>=2.0 && aperture< 4.0)
  {
    if( fabs(imgdepth[i][j]-focalplane)<20.0)
    {    sfinal.val[0]=s1.val[0]*0.7+0.3*s1.val[0];
        sfinal.val[1]=s1.val[1]*0.7+0.3*s1.val[1];
        sfinal.val[2]=s1.val[2]*0.7+0.3*s1.val[2];
    }
if( fabs(imgdepth[i][j]-focalplane)>=20.0&&fabs(imgdepth[i][j]-focalplane)<40.0)
    {    sfinal.val[0]=s1.val[0]*0.8+0.2*s1.val[0];
        sfinal.val[1]=s1.val[1]*0.8+0.2*s1.val[1];
        sfinal.val[2]=s1.val[2]*0.8+0.2*s1.val[2];
    }

if( fabs(imgdepth[i][j]-focalplane)>=40.0&&fabs(imgdepth[i][j]-focalplane)<80.0)
    {    sfinal.val[0]=s2.val[0];
        sfinal.val[1]=s2.val[1];
        sfinal.val[2]=s2.val[2];
    }
```

```
if( fabs(imgdepth[i][j]-focalplane)>=80.0&&fabs(imgdepth[i][j]-focalplane)<150.0)
    {    sfinal.val[0]=s3.val[0];
        sfinal.val[1]=s3.val[1];
        sfinal.val[2]=s3.val[2];
    }

if( fabs(imgdepth[i][j]-focalplane)>=150.0)
    {    sfinal.val[0]=s4.val[0];
        sfinal.val[1]=s4.val[1];
        sfinal.val[2]=s4.val[2];
    }

         tempimg.val[0]=int(imgdepth[i][j])%255;
        tempimg.val[1]=int(imgdepth[i][j])%255;
        tempimg.val[2]=int(imgdepth[i][j])%255;

  }

  else if(aperture>=4.0 && aperture< 8.0)
  {
    if( fabs(imgdepth[i][j]-focalplane)<20.0)
    {    sfinal.val[0]=s1.val[0];
        sfinal.val[1]=s1.val[1];
        sfinal.val[2]=s1.val[2];
    }
if( fabs(imgdepth[i][j]-focalplane)>=20.0&&fabs(imgdepth[i][j]-focalplane)<40.0)
    {    sfinal.val[0]=s1.val[0];
        sfinal.val[1]=s1.val[1];
        sfinal.val[2]=s1.val[2];
    }

if( fabs(imgdepth[i][j]-focalplane)>=40.0&&fabs(imgdepth[i][j]-focalplane)<80.0)
    {    sfinal.val[0]=s1.val[0];
        sfinal.val[1]=s1.val[1];
        sfinal.val[2]=s1.val[2];
    }

if( fabs(imgdepth[i][j]-focalplane)>=80.0&&fabs(imgdepth[i][j]-focalplane)<150.0)
    {    sfinal.val[0]=s2.val[0];
        sfinal.val[1]=s2.val[1];
        sfinal.val[2]=s2.val[2];
    }
```

```
    if( fabs(imgdepth[i][j]-focalplane)>=150.0)
        {    sfinal.val[0]=s3.val[0];
            sfinal.val[1]=s3.val[1];
            sfinal.val[2]=s3.val[2];
        }

             tempimg.val[0]=int(imgdepth[i][j])%255;
            tempimg.val[1]=int(imgdepth[i][j])%255;
            tempimg.val[2]=int(imgdepth[i][j])%255;


      }

     cvSet2D(imgfinal,i,j,sfinal);//set the (i,j) pixel value

      cvSet2D(dstemp,i,j,tempimg);

    }
}


 cvSaveImage("newfinal_No_FF.jpg", imgfinal);


cvNamedWindow("Image",1);
cvShowImage("Image",img);
cvShowImage("Final-Image",imgfinal);
 cvShowImage("dst1",dst1);
 cvShowImage("dst2",dst2);
 cvShowImage("dst3",dst3);
 cvShowImage("dst4",dst4);

cvWaitKey(0);
cvDestroyWindow( "Image" );
cvReleaseImage( &imgfinal);


 // ADD vig:
 int dar_loc=1000000;
 int vig_para_x=255;
 int vig_para_y=255;
 IplImage *img_vig=cvLoadImage("newfinal_No_FF.jpg",1);
IplImage *imgfinal_vig=cvLoadImage("newfinal_No_FF.jpg",1);
```

```
fstream infile_vig;
 infile_vig.open("2.txt");

 float t1_vig;
  for(int i=0;i<400;i++)
  {
     for(int j=0;j<600;j++)
         {
             infile_vig>>t1_vig;

             imgdepth[i][j]=t1_vig;

         }

 }
  infile.close();
   fstream infile2_vig;
  float t2_vig;
   infile2_vig.open("aperture.txt");
   infile2_vig>>t2_vig;
  aperture=t2_vig;
   infile2_vig.close();


    fstream infile3_vig;
  float t3_vig;
   infile3_vig.open("focalplane.txt");
   infile3_vig>>t3_vig;
  focalplane=t3_vig;
   infile2_vig.close();

  printf("focalplane=%f",focalplane);
   CvScalar s1_vig,sfinal_vig;


  cvSmooth(imgfinal_vig,imgfinal_vig,CV_BLUR,3,3,0,0);
  IplImage *dst1_vig = NULL , *dstfinal_vig = NULL;


  dst1_vig = cvCreateImage(cvSize(img_vig->width,img_vig->height),IPL_DEPTH_8U,3);
  dstfinal_vig = cvCreateImage(cvSize(img_vig->width,img_vig->height),IPL_DEPTH_8U,3);
  for(int i=0;i<img_vig->height;i++)
  {
```

```
        for(int j=0;j<img_vig->width;j++)
         {
            s1_vig=cvGet2D(img_vig,i,j); // get the (i,j) pixel value
              int mov_para=abs(200-i)+abs(300-j)    ;
              if (aperture<=1.0)
              {
                   int
mod_vig=(abs(200-i-2)*abs(200-i-2)*abs(200-i-2)*abs(200-i-2)+abs(300-j-2)*abs(300-j-2)*abs(3
00-j-2)*abs(300-j-2))/dar_loc*100;
                   sfinal_vig.val[0]= s1_vig.val[0]- mod_vig;
                   sfinal_vig.val[1]= s1_vig.val[1]- mod_vig;
                   sfinal_vig.val[2]= s1_vig.val[2]- mod_vig;
                   cvSet2D(imgfinal_vig,i,j,sfinal_vig);
              }
              if (aperture<=1.4 && aperture >1.0)
              {
                   int
mod_vig=(abs(200-i-2)*abs(200-i-2)*abs(200-i-2)+abs(300-j-2)*abs(300-j-2)*abs(300-j-2))/dar_
loc;
                   sfinal_vig.val[0]= s1_vig.val[0]- mod_vig;
                   sfinal_vig.val[1]= s1_vig.val[1]- mod_vig;
                   sfinal_vig.val[2]= s1_vig.val[2]- mod_vig;
                   cvSet2D(imgfinal_vig,i,j,sfinal_vig);

              }
              if (aperture<=2.8 && aperture >1.4)
              {

                   int
mod_vig=(abs(200-i-2)*abs(200-i-2)+abs(300-j-2)*abs(300-j-2))/dar_loc/100;
                   sfinal_vig.val[0]= s1_vig.val[0]- mod_vig;
                   sfinal_vig.val[1]= s1_vig.val[1]- mod_vig;
                   sfinal_vig.val[2]= s1_vig.val[2]- mod_vig;

                   cvSet2D(imgfinal_vig,i,j,sfinal_vig);

              }
              if (aperture<=4 && aperture >2.8)
              {

                   int mod_vig=(abs(200-i-2)+abs(300-j-2));
                   sfinal_vig.val[0]= s1_vig.val[0]- mod_vig;
                   sfinal_vig.val[1]= s1_vig.val[1]- mod_vig;
                   sfinal_vig.val[2]= s1_vig.val[2]- mod_vig;
```

```
                cvSet2D(imgfinal_vig,i,j,sfinal_vig);

            }

        }
    }

 cvSaveImage("newfin_vig.jpg", imgfinal_vig);
 cvNamedWindow("Image_vig",1);
 cvShowImage("Image_vig",img_vig);
 cvShowImage("Final-Image_vig",imgfinal_vig);

 cvWaitKey(0);
 cvDestroyWindow( "Image_vig" );
 cvReleaseImage( &imgfinal_vig);


 // UPDATA RENDERING:

 float CoCpara, transferOpt;
 transferOpt=0.1;
 int DiaFuzzy;
 float val_Blur=0.2;

 IplImage *img_UpRen=cvLoadImage("4b.jpg",1);
 IplImage *imgfinal_UpRen=cvLoadImage("4b.jpg",1);

fstream infile_UpRen;
 infile_UpRen.open("2.txt");

 float t1_UpRen;
 for(int i=0;i<400;i++)
 {
    for(int j=0;j<600;j++)
        {
            infile_UpRen>>t1_UpRen;

            imgdepth[i][j]=t1_UpRen;

        }

 }
 infile.close();
  fstream infile2_UpRen;
```

```
float t2_UpRen;
 infile2_UpRen.open("aperture.txt");
 infile2_UpRen>>t2_UpRen;
aperture=t2_UpRen;
 infile2_UpRen.close();


  fstream infile3_UpRen;
float t3_UpRen;
 infile3_UpRen.open("focalplane.txt");
 infile3_UpRen>>t3_UpRen;
focalplane=t3_UpRen;
 infile2_UpRen.close();

printf("focalplane=%f",focalplane);
 CvScalar s1_UpRen,s2_UpRen,sfinal_UpRen;


cvSmooth(imgfinal_UpRen,imgfinal_UpRen,CV_BLUR,3,3,0,0);
IplImage *dst1_UpRen = NULL , *dstfinal_UpRen = NULL;


dst1_UpRen = cvCreateImage(cvSize(img_UpRen->width,img_UpRen->height),IPL_DEPTH_8U,3);
dstfinal_UpRen =
cvCreateImage(cvSize(img_UpRen->width,img_UpRen->height),IPL_DEPTH_8U,3);


for(int i=0;i<img_UpRen->height;i++)
{
   for(int j=0;j<img_UpRen->width;j++)
    {
       s1_UpRen=cvGet2D(img_UpRen,i,j); get the (i,j) pixel value
         CoCpara=focalplane-imgdepth[i][j];
                      DiaFuzzy=5;
         int m=i-DiaFuzzy;

         while(m<0)
         {
              m++;
         }

         for(;m<=i+DiaFuzzy && m<img_UpRen->height;m++)
         {
              int n=j-DiaFuzzy;
```

```
                    while(n<0)
                    {
                        n++;
                    }

                    for(;(n<=j+DiaFuzzy && n<img_UpRen->width);n++)
                    {
                        int a=0;
                        if(imgdepth[i][j]=imgdepth[m][n])
                        {

                            sfinal_UpRen=cvGet2D(imgfinal_UpRen,m,n);

                            sfinal_UpRen.val[0]= s1_UpRen.val[0]*val_Blur +
(1-val_Blur)*sfinal_UpRen.val[0];
                            sfinal_UpRen.val[1]= s1_UpRen.val[1]*val_Blur +
(1-val_Blur)*sfinal_UpRen.val[1];
                            sfinal_UpRen.val[2]= s1_UpRen.val[2]*val_Blur +
(1-val_Blur)*sfinal_UpRen.val[2];

                            cvSet2D(imgfinal_UpRen,m,n,sfinal_UpRen);
                        }

                    }

                }
            }
        }
        cvSaveImage("newupdata.jpg", imgfinal_UpRen);

        cvNamedWindow("Image_UpRen",1);
        cvShowImage("Image_UpRen",img_UpRen);
        cvShowImage("Final-Image_UpRen",imgfinal_UpRen);
        cvWaitKey(0);
        cvDestroyWindow( "Image_UpRen" );
        cvReleaseImage( &imgfinal_UpRen);

        return 0;
}
```