Generating Video Frames with Depth Of Field and Motion Blur Effects for 3D Animation

A Thesis Presented

by

Karthik Sathyanarayana

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Computer Engineering

Stony Brook University

May 2010

Stony Brook University

The Graduate School

Karthik Sathyanarayana

We, the thesis committee for the above candidate for the

Master of Science degree, hereby recommend

acceptance of this thesis.

Dr. Murali Subbarao, Thesis Advisor
Professor, Department of Electrical & Computer Engineering

Dr. Ridha Kamoua, Second Reader
Associate Professor, Department of Electrical & Computer Engineering

This thesis is accepted by the Graduate School

Lawrence Martin
Dean of the Graduate School

Abstract of the Thesis

Generating Video Frames with Depth Of Field and Motion Blur Effects for 3D Animation

by

Karthik Sathyanarayana

Master of Science

in

Computer Engineering

Stony Brook University

2010

Researchers in the field of Image Processing and Computer Vision have been working hard to understand and resolve the problems arising due to depth of field (DOF) effects and relative motion between the scene and imaging system. The varying degradation seen in the image of an object due to its depth characteristics is called depth dependent defocus blur and the blurring effect observed in the images of moving objects is called motion blur. We have developed a set of algorithms to generate shift-variant defocus and motion blur effects and have presented the same through this thesis. Our algorithm precisely takes into account the rigid body motion of 3-D objects which includes translational and rotational motion in any arbitrary direction. Camera parameters such as aperture diameter, focal length and the location of image detector are used to calculate the blur circle radius of Point spread functions (PSFs) modeled by Gaussian and Cylindrical functions. We also describe a novel and simple method of image inpainting used for filling the missing pixels that arise due to round off errors occurred during interpolation or changes in magnification. Experiments have been carried out on a set of 3-D shapes like, but not limited to, sphere, cylinder, cone, etc. Results are documented to demonstrate the correctness of our algorithm. The methods described through this thesis can be used in generating realistic 3D animation. The results are also useful as simulated test data with known ground truth in the testing and evaluation of image and video de-blurring algorithms.

To all my mentors, for believing in my abilities,

my friends, for supporting me throughout the journey,

and my family for their love and affection.

Contents

List of Figures

List of Abbreviations

DOF	Depth of Field
SVPSF	Shift Variant Point Spread Function
SIPSF	Shift Invariant Point Spread Function
UI	User Interface
SCO-3D	Software camera for Objects with 3 Dimensions
AIT	Adaptive Interpolation Technique
MeanVIM	Mean Value Interpolation Method
MedianVIM	Median Value Interpolation Method

Acknowledgments

I believe that research can be viewed as a path for finding a solution to a well defined problem by the power of man, mind and machine. Through this thesis, I have been given the freedom and opportunity to do research. I sincerely thank my advisor, Professor Muralidhara Subbarao for his willingness to support me and allow me to present my research work through this thesis. I thank him for the motivation, encouragement and guidance he has constantly provided without which, this work would not have been possible.

I am also grateful to Professor Ridha Kamoua for serving on my thesis committee and reviewing it.

I would like to thank all current and previous members of the Computer Vision Lab: Mr. Shekhar Sastry, Mr. Younsik Kang and Dr. Xue Tu. It has been a great pleasure to work with them and discuss several research problems. I would like to convey my special thanks to Mr. Shekhar Sastry, who has constantly guided me and helped me out during the course work of my thesis.

I express my sincere thanks to Professor Nagaraj Ramarao for supporting me during my initial days of research in the field of Image Processing.

And special thanks goes to my parents for supporting me unconditionally in the past years.

I would finally like to thank everyone who have believed in my abilities and more importantly those who have challenged them.

Chapter 1

Introduction

Blurring effects in an image can arise due to depth of field (DOF) and relative motion between imaging system and the object being imaged. Such effects on one hand are problematic for computer vision algorithms that try to extract details from an image. And on the other hand they are useful for generating highly realistic 3D animation. At present, 3D animation video generated in computer graphics applications does not seem to include realistic depth-of-field effects (i.e. depth dependent defocus blur) and motion blur effects that are associated with human vision as well as actual video cameras. Introducing such effects would enhance the perception of animation video to be more realistic and natural.

In this thesis we demonstrate a set of algorithms to generate such images with shift variant defocus and motion blur. We consider a set of 3D shapes such as cylinder, cone, sphere, ellipsoid, etc for our experiments. Rigid body motion of 3-D objects which includes translational and rotational motion in any arbitrary direction is taken into account. Camera parameters such as aperture diameter, focal length and the location of image detector are used to calculate the blur circle radius of Point spread functions (PSFs) modeled by Gaussian and Cylindrical functions. We present a novel and simple scheme for filling the missing pixels that occur during the generation of such images due to round off errors and magnification effects. This scheme for addressing missing pixels is extended as a solution to the problem of Image Inpainting.

We have developed a software program titled "Software Camera for Objects with 3 Dimensions (SCO-3D)" to demonstrate our algorithms and view the results. Our software allows the user to set the various camera, shape and motion parameters through an intuitive user interface. The software is capable of producing video frames with shift variant defocus and motion blur based on the specified object and motion parameters. SCO-3D can be used as a research tool to generate test data for the verification of image and video deblurring algorithms. The images generated by our software can be compiled to result in animation video that shows depth of field and motion blur effects as seen by human vision or as captured by a real video camera. We can also use SCO-3D to generate stereo images of different 3D shapes with shift variant defocus and motion blur effects. These images can be used for research in the field of shape from defocus, shape from motion and shape from stereo.

1.1 Thesis overview

In Chapter 2 we describe the fundamentals of imaging such as pin hole camera model and the image blurring process. We derive the equations for shift variant and shift invariant blurring which are used to generate images of 3D objects in subsequent chapters. We also list the rigid body motion (translation and rotation) formulae that will be used to simulate object motion in any arbitrary direction.

In Chapter 3 we choose a set of preliminary 3D shapes such as cylinder, sphere, cone, plane and ellipsoid to generate images with shift variant defocus blur. For generating such images we first derive the equations to determine the depth of all visible points of a 3D shape relative to the lens of a camera. Using these equations and camera parameters we determine the amount of blurring in the image based on standard PSF models such as Cylindrical and Gaussian. We demonstrate the results generated by our software SCO-3D.

In Chapter 4 we present our method to generate images of moving 3D objects. The images that we generate contain shift variant defocus and motion blur. We resolve the problem of missing pixels that occurs during the generation of such images using a heuristic algorithm.

In Chapter 5 we extend our algorithm for the problem of missing pixels to address the more general problem of image inpainting. Here, we show the results of our method on a set of natural and synthetically generated test images.

In Chapter 6 we present the summary of our work that highlights its importance. Finally, we discuss the future scope of our research.

Chapter 2

Imaging Fundamentals

2.1 Motivation

The image of a planar object placed perpendicular to the optical axis facing the lens such that it is not in focus, will contain the same amount of blur at every pixel based on geometic optics. This is also called as shift invariant blur. However, the image of a 3D object using a limited depth of field imaging system will contain varying amount of blur at every pixel. This is also called as shift variant blur. Through this chapter we try to understand the optics and derive mathematical relations associated with the process of blurring.

This chapter describes three important aspects of imaging required to understand the algorithm used for generating images with shift variant blur and motion blur. Firstly, we explain the pin hole camera model and derive a relation between the object point and corresponding image point. Next, we state the equations associated with rigid body motion which will be useful for simulating object motion in 3 dimensions. Finally, we explain the image blurring process and the equations associated with both types of blur namely, shift invariant and shift variant.

2.2 Camera model and perspective projection

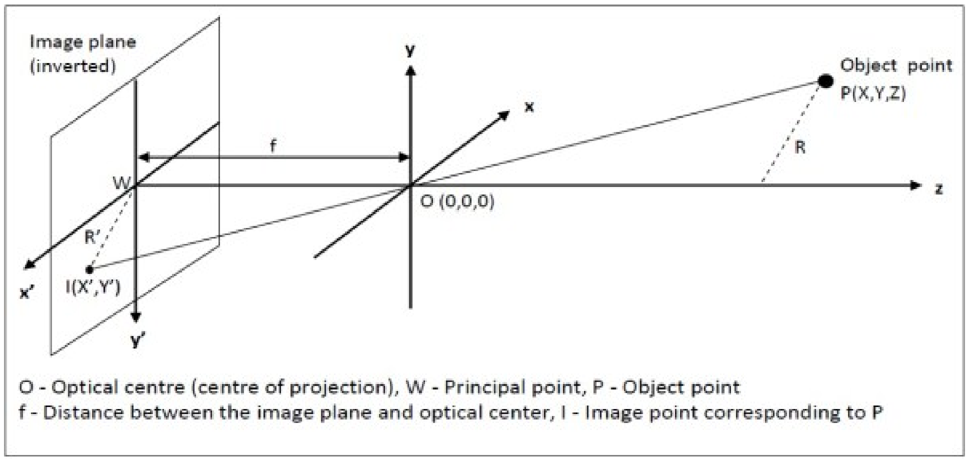


Image plane (inverted)
y
Object point P(X,Y,Z)
x
f
R
W
z
O (0,0,0)
R'
x'
I(X',Y')
y'
O - Optical centre (centre of projection), W - Principal point, P - Object point
f - Distance between the image plane and optical center, I - Image point corresponding to P

Figure 2.1: Pin hole camera model

We use the pin hole camera model to understand the geometry of the image formation process as shown below in Figure 2.1.
The projection of the object point P onto the image plane is obtained by locating the point of intersection of the line OP with the image plane when extended towards the image plane. Using this model we derive a relation between the 3D coordinates of the object point and the image point. To avoid the inverted image problem we place the image plane at a distance f in front of the optical centre as shown in Figure 2.2.

I(X',Y') is the image point corresponding to object point P. We observe that the distance of point P from the **z** axis is $R = \sqrt{X^2 + Y^2}$ and the distance of point I

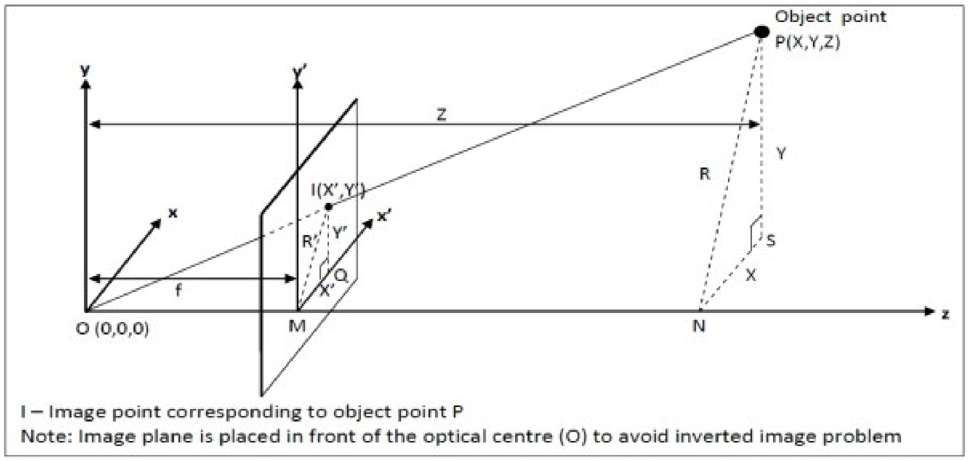

Object point
P(X,Y,Z)
y
Y'
Z
Y
R
I(X',Y')
x
x'
R'
Y'
S
X
Q
f
X'
O (0,0,0)
M
N
z
I – Image point corresponding to object point P
Note: Image plane is placed in front of the optical centre (O) to avoid inverted image problem

Figure 2.2: Perspective projection model [1]

from the **z** axis is $R' = \sqrt{X'^2 + Y'^2}$. Triangle IMO and triangle PNO are similar triangles and thus we have the following relation:

$$\frac{f}{ON} = \frac{R'}{R} \tag{2.1}$$

Triangles IQM and PSN are also similar triangles. Thus we can have the relations:

$$\frac{X'}{X} = \frac{Y'}{Y} = \frac{R'}{R} \tag{2.2}$$

Combining equations 2.1 and 2.2 we obtain the following: For convenience we denote the distance ON by z

$$\frac{X'}{X} = \frac{f}{z} \qquad and \qquad \frac{Y'}{Y} = \frac{f}{z} \tag{2.3}$$

Thus the coordinates of the point P(X,Y,Z) on the image plane is given by the equations

$$X' = \frac{f}{z}X \tag{2.4}$$

$$Y' = \frac{f}{z}Y \tag{2.5}$$

We will use the equations 2.4 and 2.5 later in Chapter 3 where we explain the algorithm and equations to generate images with shift variant defocus blur and motion blur.

2.3 Rigid body motion

To determine the new position of an object/scene point after translational and/or rotational motion we use the rigid body motion formulae described in this section [1]. By definition, rigid body motion assumes that there is no change in the shape or size of an object during its motion. Translational motion can be described by considering velocity of the object point in x, y and z direction with reference to camera coordinates. Rotational motion can be characterized by the knowledge of angular velocity about the x, y and z axis with respect to the object coordinates. The origin of the camera coordinates is considered as the centre of the lens and also called the Optical center, see Figure 2.2. The object coordinate system has its origin at the center of the object. As an example, for a spherical object, its center is considered as the origin of the object coordinate system. We will define the object center in detail in Chapter 3.

A brief description of the important parameters of translational and rotational motion are as follows:

X_0, Y_0, Z_0	-	The x, y and z coordinates of the initial position of the object point in camera coordinate system.
v_x, v_y, v_z	-	The velocity of the object in x, y and z directions respectively.
X_1, Y_1, Z_1	-	The x, y and z coordinates of the object point after translational motion with reference to camera coordinate system.
ω_x, ω_y, ω_z	-	The angular velocity of object points about the x, y and z axis of the object coordinate system.
θ_x, θ_y, θ_z	-	The angle made by an object point about the x, y and z axis respectively after rotational motion with respect to the object coordinates.
X', Y', Z'	-	The final position of the object point after rotational and translational motion as per camera coordinates.
t	-	The time interval between 2 consecutive frames of a video that captures moving objects.

The relation between new position of the object point after translational motion is given by the following equations :

$$X_1 = X_0 + v_x t \tag{2.6}$$

$$Y_1 = Y_0 + v_y t \tag{2.7}$$

$$Z_1 = Z_0 + v_z t \tag{2.8}$$

The equations depicting the new angles made by the object points after rotational motion only are :

$$\theta_x = \omega_x t \tag{2.9}$$

$$\theta_y = \omega_y t \tag{2.10}$$

$$\theta_z = \omega_z t \tag{2.11}$$

The relation between the new position (X',Y',Z') of an object point initially at (X_0,Y_0,Z_0) after translational and rotational motion with parameters specified above is given by

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} X_0 + v_x t \\ Y_0 + v_y t \\ Z_0 + v_z t \end{bmatrix} + \begin{bmatrix} \\ \mathbf{R} \\ 3 \times 3 \textit{ rotation matrix} \end{bmatrix} \begin{bmatrix} X_1 \\ Y_1 \\ Z_2 \end{bmatrix} \tag{2.12}$$

where Z_2 is the distance of the object point after translational motion from the xy-plane in the object coordinate system. It can be computed using the equation below

$$Z_2 = \textit{(z-coordinate of the object center in camera coordinate system)} - Z_1 \tag{2.13}$$

The 3×3 orthonormal matrix for rotation $\mathbf{R}$ is given by

$$\mathbf{R} = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix} \tag{2.14}$$

The entries of the rotation matrix R are defined in the equations below

$$r_{xx} = cos(\theta_y)cos(\theta_z) \tag{2.15}$$

$$r_{xy} = sin(\theta_x)sin(\theta_y)cos(\theta_z) + cos(\theta_x)sin(\theta_z) \tag{2.16}$$

$$r_{xz} = -cos(\theta_x)sin(\theta_y)cos(\theta_z) + sin(\theta_x)sin(\theta_z) \tag{2.17}$$

$$r_{yx} = -cos(\theta_y)sin(\theta_z) \tag{2.18}$$

$$r_{yy} = -sin(\theta_x)sin(\theta_y)sin(\theta_z) + cos(\theta_x)cos(\theta_z) \tag{2.19}$$

$$r_{yz} = cos(\theta_x)sin(\theta_y)sin(\theta_z) + sin(\theta_x)cos(\theta_z) \tag{2.20}$$

$$r_{zx} = sin(\theta_y) \tag{2.21}$$

$$r_{zy} = -sin(\theta_x)cos(\theta_y) \tag{2.22}$$

$$r_{zz} = cos(\theta_x)cos(\theta_y) \tag{2.23}$$

2.4 Image blurring process

The Figure 2.3 shows the image formation process using paraxial geometric optics. For a convex lens, the light radiated by the object point P, intercepted by the lens, is refracted by the lens to converge at a point P' on the image plane (IP). The distance of the point P can be determined by the well known Lens maker's formula

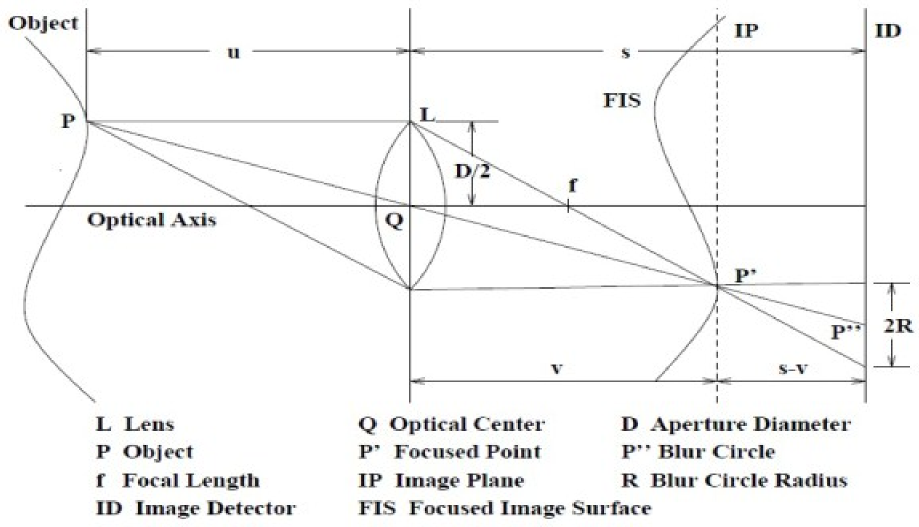


Figure 2.3: Image formation and blurring process [2]

below.

$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f} \tag{2.24}$$

where, u = distance between the lens and the object point P, v = distance between lens and image point P' and f = focal length of lens.

If the image detector is moved to coincide with the image plane (IP) then a focused image of the object point P is obtained. A point object at a distance u on one side of the lens produces a point image at a distance v on the other side of the lens with focal length f. If the image detector (ID) is displaced relative to the image plane by a distance s - v, such that it still remains parallel to the image plane, then the point object P no more produces a point image on the image detector. The light from the object point P gets distributed over a circular region on the image detector. The circular region is called blur circle with radius R, see Figure 2.3. The distribution of light in a blur circle is given by the 2D Point Spread Function (PSF) which is denoted as $h(x, y)$. The PSF can be modelled by using geometric optics or wave optics. Although, the physical distribution of light as modelled by wave optics would be more accurate, it is sufficient to assume Gaussian or Cylindrical PSF for our purpose. A detailed analysis of the different PSFs using wave optics is given in [4]. We restrict our PSF models to be either Gaussian or Cylindrical in shape. For most aberration free optical systems Gaussian PSF is regarded as a good assumption.

Assuming that the camera system is a lossless system we have

$$\int\int h(x, y)\, dxdy = 1 \tag{2.25}$$

The cylindrical PSF can be represented as

$$h(x, y) = \begin{cases} \frac{1}{\pi R^2} & \textit{if } x^2 + y^2 \leq R^2 \\ 0 & \textit{otherwise} \end{cases} \tag{2.26}$$

Here, the light from object point P is assumed to have a uniform distribution within the circular region or the blur circle.

A more accurate model for the PSF would be the Gaussian model which can be represented as

$$h(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.27}$$

where σ is a spread parameter corresponding to the standard deviation of

Gaussian distribution.

In practice, it is found that σ is proportional to the blur circle radius Equation 2.33 i.e.

$$\sigma = kR' \text{ for k} > 0 \tag{2.28}$$

In most practical cases

$$k = \frac{1}{\sqrt{2}} \tag{2.29}$$

is a good approximation [5, 6].

For all practical purposes we assume that the area under Gaussian, effective for calculating PSF coefficients is a circular region with radius 2σ from the peak. We normalize the PSF coeffients to maintain the condition in Equation 2.25.

The blur circle is observed for both positive and negative displacements of image detector. We take into consideration that the object point is defocused on the image detector only if the blur circle radius R $\geq$ 0.5.

Next, we derive a relation to express the blur circle radius in terms of known parameters of the optical system using geometric optics. Let R = Radius of blur circle, D = Diameter of the lens aperture, s = Distance between the image detector and the lens. From similar triangles in Figure 2.3 we have

$$\frac{2R}{D} = \frac{s - v}{v} = s\left(\frac{1}{v} - \frac{1}{s}\right) \tag{2.30}$$

Using the value of $\frac{1}{v}$ from Eq.2.24 we have

$$\frac{2R}{D} = s\left(\frac{1}{f} - \frac{1}{u} - \frac{1}{s}\right) \tag{2.31}$$

Therefore,

$$R = \frac{sD}{2}\left(\frac{1}{f} - \frac{1}{u} - \frac{1}{s}\right) \tag{2.32}$$

R can be positive or negative depending on the position of the image detector being towards or away from the lens with reference to image plane. We introduce a magnification normalization factor corresponding to $s = s_0$. The normalized blur circle radius is given as R' = $\frac{s_0 R}{s}$. Thus Eq. 2.32 can be written as

$$R' = \frac{s_0 D}{2}\left(\frac{1}{f} - \frac{1}{u} - \frac{1}{s}\right) \tag{2.33}$$

A brief discussion on magnification normalization can be seen in [7].

We will use Eq.2.33 in Chapter 3 to determine the blur circle radius for every object point in a 3D scene.

We now derive a relation between the focussed image and the blurred image using linear system theory. We can express the shift variant blurred image in terms of the focused image and noise component as follows

$$g(x, y) = H[f(x, y)] + \eta(x, y) \tag{2.34}$$

where g(x,y) = intensity of the blurred image at (x,y) , f(x,y) = intensity of focused image at point (x,y) , $\eta(x, y)$ = is the random noise and $H[]$ is the linear blurring operator. The focused image can be represented as

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\alpha, \beta)\delta(x - \alpha, y - \beta)\, d\alpha\, d\beta \tag{2.35}$$

where $\delta(x-\alpha, y-\beta)$ is the Dirac delta or point object function. Using Eq.2.35 in Eq. 2.34 we can write the expression for blurred image ignoring the noise component as follows:

$$g(x, y) = H[f(x, y)] = H[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\alpha, \beta)\delta(x - \alpha, y - \beta)\, d\alpha\, d\beta] \tag{2.36}$$

Using the property of linear operator H we get

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} H[f(\alpha, \beta)\delta(x - \alpha, y - \beta)\, d\alpha\, d\beta] \tag{2.37}$$

As $f(\alpha, \beta)$ is independent of x and y we can rewrite the Eq.2.37 as

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\alpha, \beta)H[\delta(x - \alpha, y - \beta)\, d\alpha\, d\beta] \tag{2.38}$$

The term

$$h(x, \alpha, y, \beta) = H[\delta(x - \alpha, y - \beta)] \tag{2.39}$$

is called as the 2D Shift Variant Point Spread Function (SVPSF) of the optical system. This means that the PSF at every point on the image varies depending on the depth and the position of the object point.

Thus, we can express the blurred image $g(x, y)$ in terms of the focused image

$f(x, y)$ and the 2D SVPSF $h(x, \alpha, y, \beta)$ as shown below

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\alpha, \beta) h(x, \alpha, y, \beta) \, d\alpha \, d\beta \tag{2.40}$$

Eq. 2.40 is also called Fredholm integral of the first kind.

In case of the shift invariant blur we can represent the blurred image as follows :

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\alpha, \beta) h(x - \alpha, y - \beta) \, d\alpha \, d\beta \tag{2.41}$$

This expression represents the convolution integral. A good descripton of the related theory is given in [8].

In further chapters we will use the concepts and equations related to shift variant blurring for generating realistic synthetic images of 3D objects.

Chapter 3

Generating Images with Shift-Variant Defocus Blur for Stationary Objects

3.1 Introduction

In Chapter 2 we derived the relation between focused image of an object and its corresponding blurred image under the shift invariant (Equation 2.41) and shift variant blurring model (Equation 2.40). In this chapter we present an algorithm, related expressions and results for producing shift variant blurred images of objects with different shapes under known conditions. We use the Gaussian PSF model in most of our experiments due its the smooth decay.

The chapter begins with a literature survey where we talk about related work. We then introduce our algorithm along with the assumptions and steps involved to compute the images of different shapes. Next, we derive the expression for depth 'Z' of object points based on the object shape. We also provide experimental results (images) for each object shape listed below. Finally, we provide a generalized method for computing the image of any quadratic surface.

The different shapes of the objects considered in our experiments are

- Planar
- Cylindrical
- Spherical
- Conical
- Elliptical

3.2 Literature survey

Michael et al. [9] describe a method to produce synthetic images with shift variant blur. Using a table of sampled values of Z (distance between object point and the lens) and corresponding PSFs they calculate the blur at every pixel. Subbarao et al. [10] have demostrated a method where they simulate the formation of defocused images as taken by a CCD camera.

3.3 Algorithm to generate synthetic images of 3D shapes

In this section we explain the algorithm to generate shift variant blurred images of stationary 3D objects (except for Planar being 2D) and provide resultant images with test data.

3.3.1 Assumptions

For stationary 3D objects we work under the following assumptions:

- The centre of all objects under consideration lie on the optical axis (This will be more clear in the ray diagram corresponding to every object given in further sections).
- The optical medium does not introduce any degradation effect on the light passing through it.
- All shading and shadow effects in the image of the object are ignored.
- Object sizes are chosen in a manner as to restrict the maximum blur circle radius to (around) 6 pixels. This limits the time required for the computation of blurred image in software.
- We do not take into consideration the diffraction effects which are caused due to finite aperture size.

3.3.2 Input Parameters

The algorithm requires the following values as input.

- Camera Parameters
  - Focal length of lens - f
  - Aperture diameter - D
  - Pixel size - p
  - Distance between image detector and lens - s
  - Magnification normalization factor - s_0 (assumed to be equal to f)

    Note: All parameters are reduced to the same scale.
- Object shape - Planar/Cylindrical/Spherical/Conical/Elliptical
- Object shape parmaters - This will be listed for every object in later subsections.
- Test image is considered as the input focused image of the object. The focused image is used as the texture for the object under consideration.

3.3.3 Blurred image computation

The steps involved in computing the blurred image of an object can be listed as follows:

1. Read the test image and store its intensity component values into a 2D array.
2. Choose a pixel at location [i,j] within the 2D array and compute its position (x,y) in reference to the axis passing through the centre pixel as shown in Fig 3.2 using the Equation 3.1 and Equation 3.2

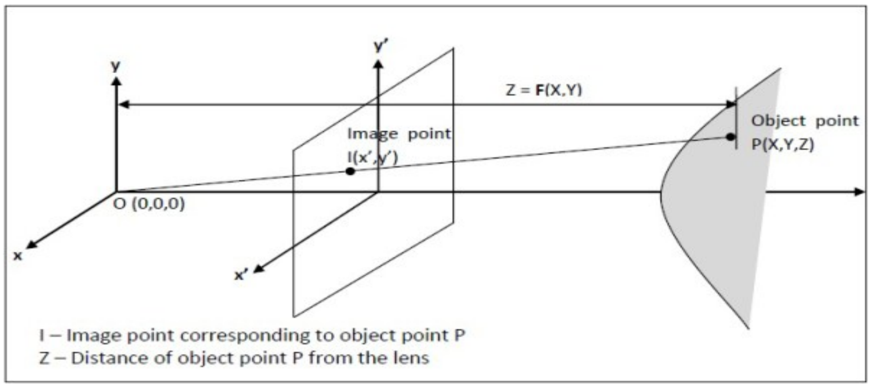

y
y'
Z = F(X,Y)
Image point
I(x',y')
Object point
P(X,Y,Z)
O (0,0,0)
x
x'
I – Image point corresponding to object point P
Z – Distance of object point P from the lens

Figure 3.1: Image formation ray diagram

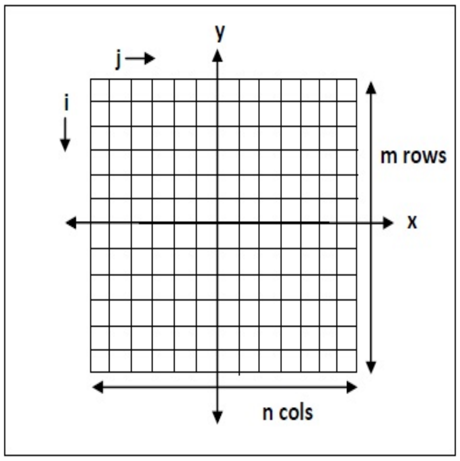


Figure 3.2: Representation of the image detector with m rows and n columns. [i,j] is the 2D array index notation for a pixel whereas (x,y) is the 2D coordinates of the centre of a pixel

$$x = p\left((j-1) - \left(\frac{n-1}{2}\right)\right) \tag{3.1}$$

$$y = p\left(-(i-1) + \left(\frac{m-1}{2}\right)\right) \tag{3.2}$$

3. Compute the Z value (depth) using shape information and perspective

projection formulae (see Equation 2.4 and Equation 2.5). We will derive the general expression to compute Z value for every shape listed in section 3.1 going further in this chapter. A generic correspondence diagram is given Figure 3.1.

4. With the depth information i.e. Z value of each pixel we calculate the blur circle radius R' corresponding to it using Equation 2.33.

   For Gaussian PSF we compute the value of σ using the equation below

$$\sigma = \sqrt{2}R' \tag{3.3}$$

5. For every object point the PSF coefficients are computed using Equations 2.26 or 2.27 and are normalized. Next, we distribute the intensity of the chosen pixel in the focused image onto the blurred image based on the PSF.

6. Repeat the above steps for every pixel in the focused image to compute the blurred image effectively implementing the Equation 2.40 also called as Fredholm Integral of the first kind.

The common test image of a brick wall for our experiments is shown in Fig 3.3 The default values of the lens parameters in our experiments were as follows:

- Focal length of the lens - $f = 20.0$ mm
- Aperture diameter - $D = 10.0$ mm
- Pixel size - $p = 0.01$ mm
- Distance of Image plane from lens - $S = 21.739$ mm

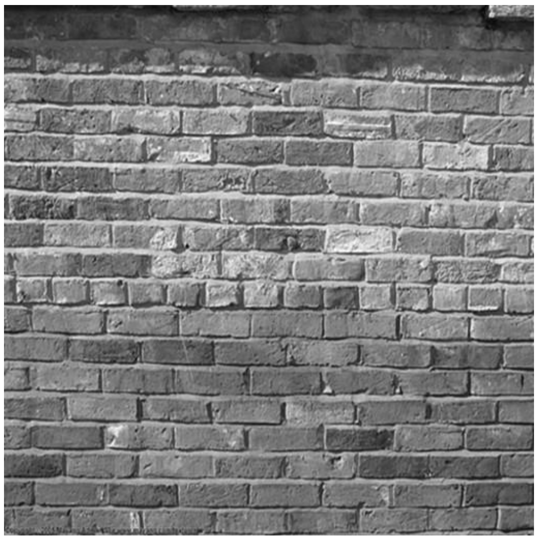


Figure 3.3: Test image that will be used as texture for all shapes

- Magnification normalization factor - $S_0 = 20.0$ mm

As per the above values all object points with a depth of field centered at 250 mm will be focused in the image.

3.4 Challenges in implementing the algorithm

1. During the generation of such images we need to efficiently take care of occlusion effects and conditions under which any part of the object is not within the scope of the camera. We resolve the occlusion problem by considering the pixel closest to the lens when compared to overlapping pixels farther away from the lens. Parts of the object which are out of scope of the camera are ignored during processing.

2. The Z value (depth) determined can result in a complex number in which case, that pixel in the focused image does not contribute to an object point. It plays

the role of a background pixel with intensity 0.

3. We stated before that we will be using the Gaussian PSF model in most of our experiments. This is because prominent quantization error and round off error in case of cylindrical PSF introduces discontinuity at the pixels separating the regions with different blur circle radii. However, due to smooth change in the Gaussian PSF compared to a step function resemblence of a cylindrical PSF we do not face such issues.

3.5 Software Camera for Objects in 3 Dimensions (SCO-3D)

At the Computer Vision Laboratory, Department of Electrical and Computer Engineering, Stony Brook University, we have developed a software titled "Software Camera for Objects in 3 Dimensions (SCO-3D)" in MATLAB supported by an intuitive UI. This software allows the user to easily select the required input parameters associated with the camera and object shape to generate images with shift variant defocus blur and motion blur. Further details of the software and selected source code are provided in the appendix section of the thesis.

3.6 Shape based depth estimation of object points

For all object shapes listed in section 3.1 we derive a closed form expression for the value of Z (depth) of the object points. Z will be expressed in terms of known parameters such as position of the corresponding focussed image pixel, shape constants (such as slope of a plane, radius of a sphere, etc) and focal length of the lens.

We can express the depth Z of an object point as a function of its x and y coordinates (see Fig 3.1) based on its geometry i.e.

$$Z = F(X, Y) \tag{3.4}$$

Substituting the value of X and Y using perspective projection relations (see Equation 2.4 and Equation 2.5) we rewrite the Equation 3.4 as

$$Z = F\left(\frac{xZ}{f}, \frac{yZ}{f}\right) \tag{3.5}$$

3.6.1 Planar object

The standard equation of a plane in 3D is given by

$$BX + CY + DZ + A = 0 \tag{3.6}$$

We can rewrite the above equation representing Z in the rest of the terms as

$$Z = a + bX + cY \tag{3.7}$$

where $a = \frac{-A}{D}$ denotes the distance between the centre of the plane and the lens, $b = \frac{-B}{D}$ denotes the slope along x axis and $c = \frac{-C}{D}$ denotes the slope along y axis.

Using perspective projection formulae we get

$$Z = a + b\left(\frac{xZ}{f}\right) + c\left(\frac{yZ}{f}\right) \tag{3.8}$$

Grouping terms with Z we get

$$Z\left(1 - \frac{bx}{f} - \frac{cy}{f}\right) = a \tag{3.9}$$

Thus, a closed form equation for Z in terms of x,y and f is given by

$$Z = \frac{a}{\left(1 - \frac{bx}{f} - \frac{cy}{f}\right)} \tag{3.10}$$

The results based on our algorithm to generate images of a planar object with texture as shown in Fig 3.3 are as shown in Fig 3.4 and Fig 3.5.

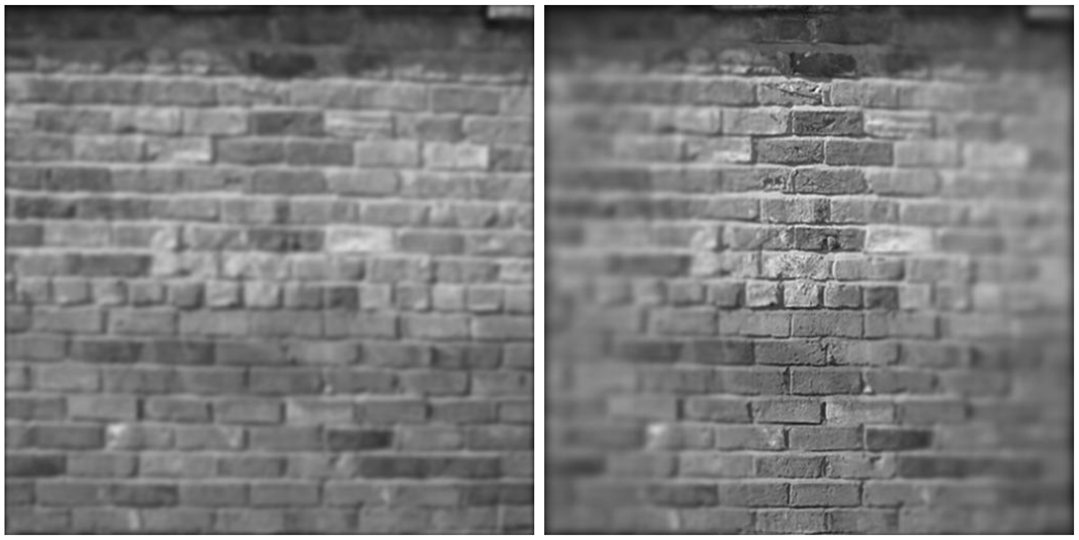


Figure 3.4: Left: Image of a planar object parallel to the lens which is out of focus i.e. $b = 0$, $c = 0$ and $a = 230$ mm, Right: Planar object titled along y axis i.e. $b = 1.5$, $c = 0$ and $a = 250$ mm

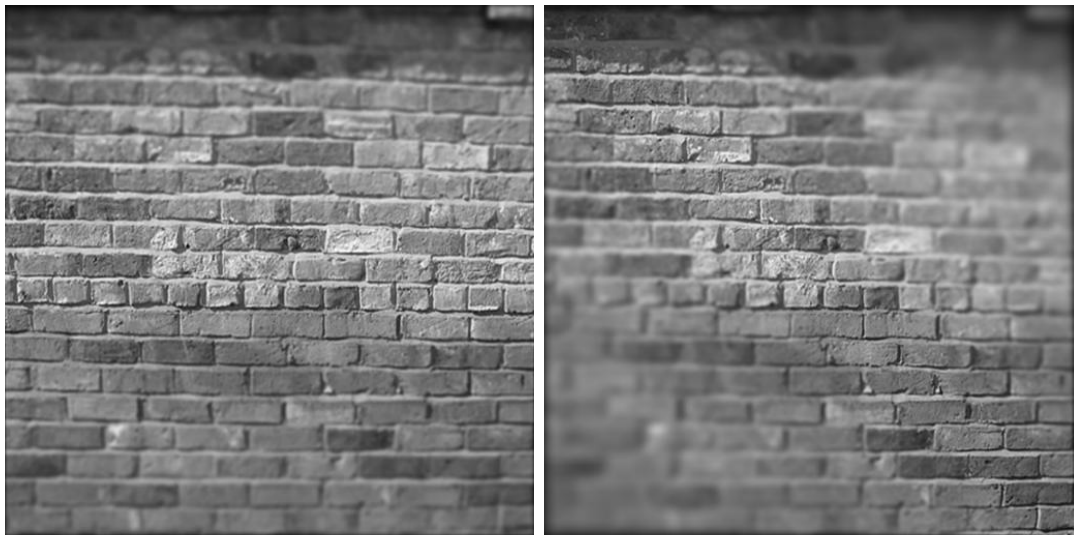


Figure 3.5: Left: Planar object titled along x axis i.e. $c = 0.7$, $b = 0$, $a = 250$ mm, Right: Planar object titled along both x and y axis i.e. $c = 1$, $b = 1$, $a = 250$ mm

Figure 3.4 left image shows the image of a planar object that is out of focus and with shift invariant blur. This implies all object points are at the same depth from the lens.

Figure 3.4 right image shows the image of a tilted planar object. The image contains shift variant blur such that the amount of blur is increasing as the object points move away from the y axis or in a direction perpendicular to the y axis.

Figure 3.5 left image shows the image of a tilted planar object. The image contains shift variant blur such that the amount of blur is increasing as the object points move away from the x axis.

Figure 3.5 right image shows the image of a tilted planar object. The image contains shift variant blur such that the amount of blur is increasing as the object points move away from the main diagonal.

3.6.2 Cylindrical object

In this section we derive the expression for the distance 'Z' of any point on a right circular cylindrical object from the lens as shown in Fig 3.6 and 3.7. The cylindrical object is assumed to be aligned such that the line passing through the centre of the cylinder parallel to its surface is perpendicular to the optical axis. The shape specific constant for a right circular cylindrical object is the radius ρ and the position of the centre 'C' of the object (see Fig 3.6).

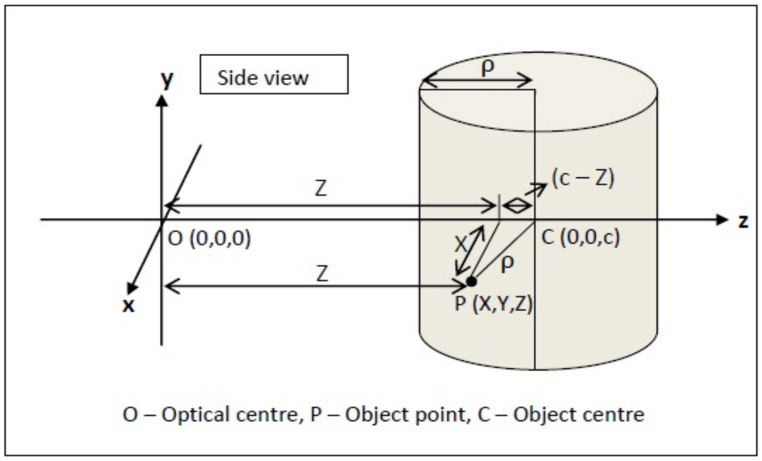


Figure 3.6: Ray diagram illustrating the side view geometry associated in imaging a cylindrical object with radius ρ

Using the property of right angled triangle CPR in Fig 3.7 we can write the expression for radius as

$$\rho^2 = X^2 + (c - Z)^2 \tag{3.11}$$

Rearranging the terms and substituting the value of X using perspective projection formulae we get

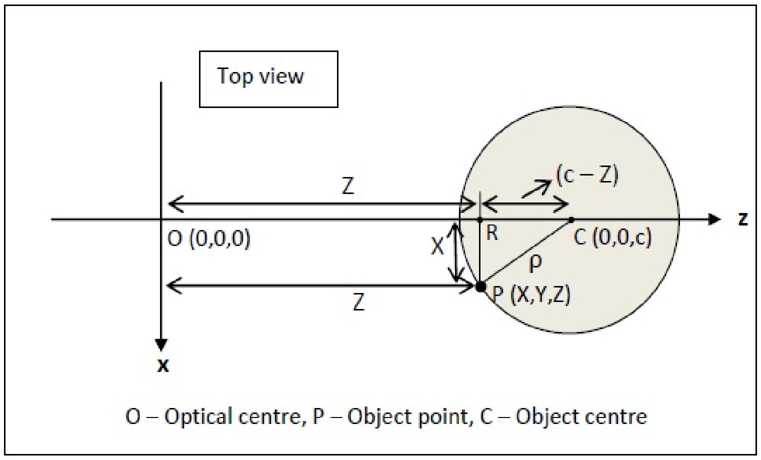


Figure 3.7: Ray diagram illustrating the top view geometry associated in imaging a cylindrical object with radius ρ

$$(c - Z)^2 + \left(\frac{xZ}{f}\right)^2 = \rho^2 \tag{3.12}$$

Further we reduce equation 3.12 to derive a quadratic equation in Z as follows

$$c^2 - 2cZ + Z^2 + \left(\frac{x^2 Z^2}{f^2}\right) = \rho^2 \tag{3.13}$$

$$\left(1 + \frac{x^2}{f^2}\right) Z^2 - 2cZ + (c^2 - \rho^2) = 0 \tag{3.14}$$

Denoting $a' = \left(1 + \frac{x^2}{f^2}\right)$, $b' = -2c$ and $c' = c^2 - \rho^2$ we can write the quadratic equation in Z as

$$a'Z^2 + b'Z + c' = 0 \tag{3.15}$$

The general solution for Equation 3.15 is given by

$$Z = \frac{-b' \pm \sqrt{(b'^2 - 4a'c')}}{2a'} \tag{3.16}$$

The above Equation 3.16 represents the closed form equation for the value of Z for every point on the surface of a cylindrical object shown in Fig 3.6 and 3.7. We choose only positive and real values of Z.

An important characteristic of a cylindrical object is that, all points with the same x coordinate (according to the object coordinate system) on the surface of the cylinder will be at the same distance from the lens irrespective of their y coordinate. This implies that the blur in the image will change only along the x direction for an arrangement as shown in Fig 3.6.

The results based on our algorithm to generate images of a cylindrical object with texture as shown in Fig 3.3 are shown in Fig 3.8 and 3.9.

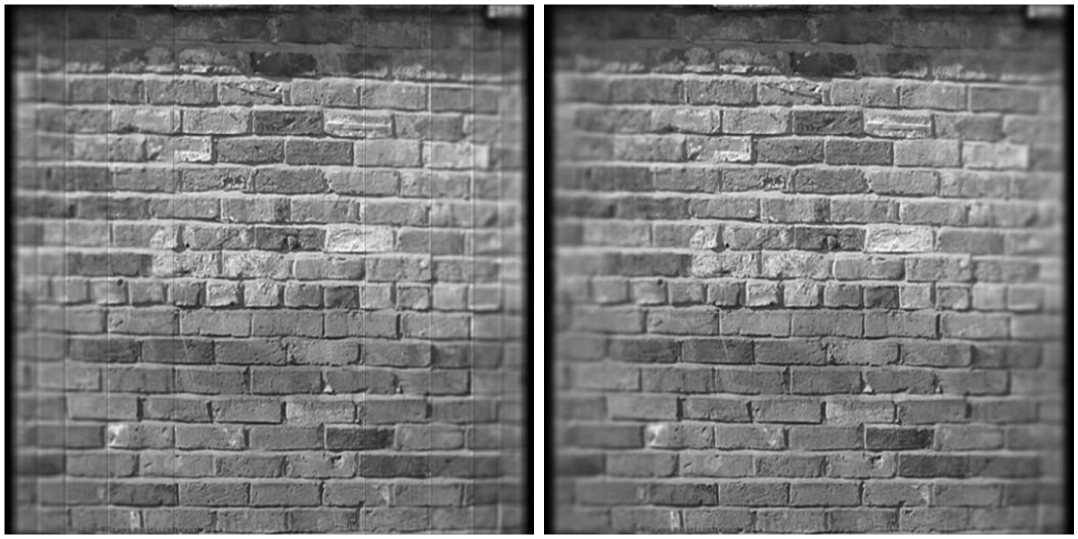


Figure 3.8: Left: Image of cylindrical object under cylindrical PSF model. Right: Image of cylindrical object under Gaussian PSF model

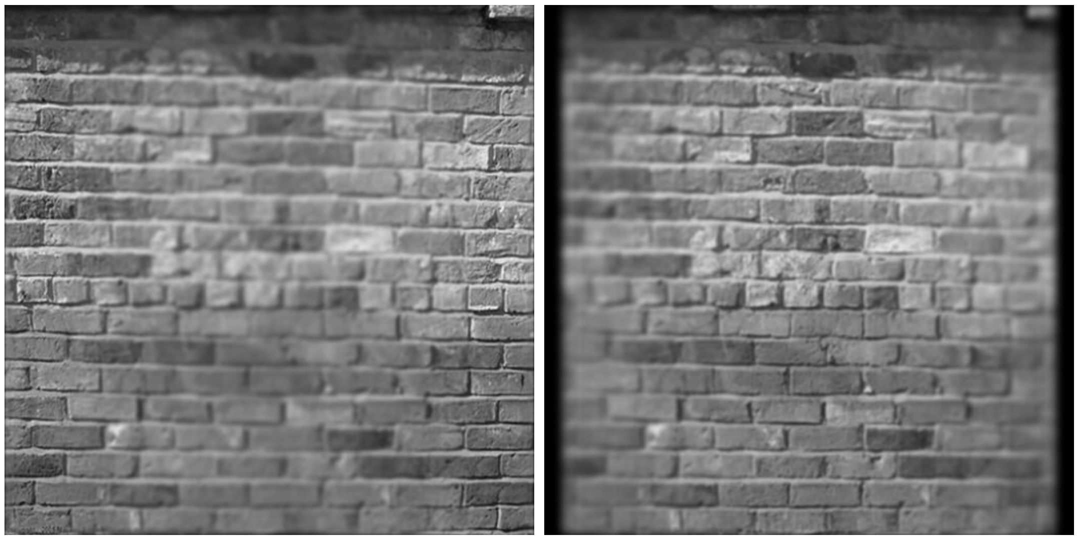


Figure 3.9: Images of cylindrical object with depth dependent defocus blur

Fig 3.8 on the left shows the image of a circular cylindrical object using a cylindrical PSF model. We can clearly observe the increase in blur as the object surface moves away from the lens. The position of the cylinder has been chosen such that the central area is in focus.

Fig 3.8 on the right shows the image of a circular cylindrical object using a Gaussian PSF model. The rest of the conditions remain same as the left image.

Fig 3.9 shows the images of the cylindrical object with the centre of the cylinder placed at varying distances from the lens to illustrate the effect of depth dependent defocus blur.

3.6.3 Spherical object

In this section we derive the expression for the distance 'Z' of any point on a spherical object whose centre lies on the optical axis as shown in Fig 3.11 and 3.10.

The shape specific constant for a spherical object is the radius ρ and the position

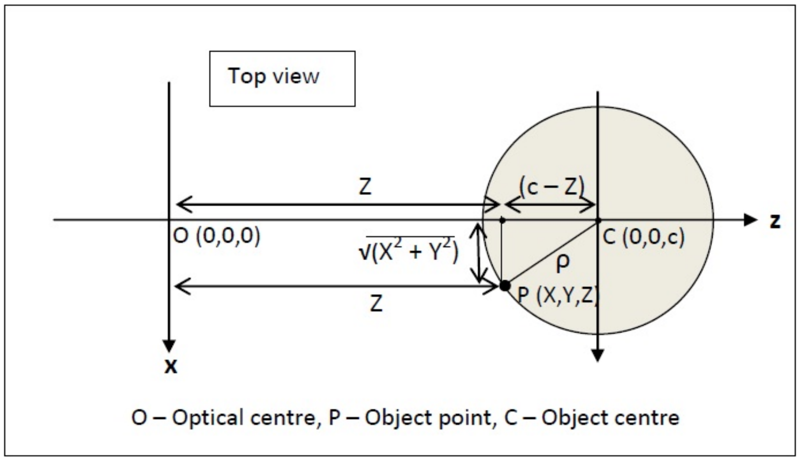


Figure 3.10: Ray diagram illustrating the top view geometry associated in imaging a spherical object with radius ρ

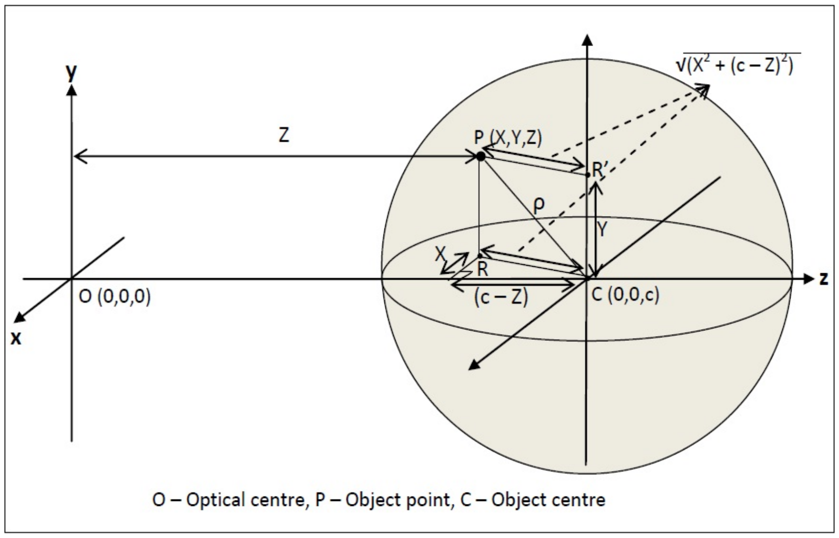


Figure 3.11: Ray diagram illustrating the geometry associated in imaging a spherical object with radius ρ

of its centre C (see Fig 3.11).

The general equation of a sphere with radius ρ and centre at (i, j, k) is given by

$$(x - i)^2 + (y - j)^2 + (z - k)^2 = \rho^2 \tag{3.17}$$

From Fig 3.11 right angled triangles CRP and PR'C are congruent. Thus, we have the relation

$$X^2 + (c - Z)^2 + Y^2 = \rho^2 \tag{3.18}$$

Using the perspective projection relations and rearranging the terms we get

$$\left(\frac{x^2 + y^2}{f^2}\right) Z^2 + c^2 + Z^2 - 2cZ = \rho^2 \tag{3.19}$$

Reducing further we have

$$\left(\frac{x^2 + y^2}{f^2} + 1\right) Z^2 - 2cZ + (c^2 - \rho^2) = 0 \tag{3.20}$$

Denoting $a' = \left(1 + \frac{x^2+y^2}{f^2}\right)$, $b' = -2c$ and $c' = c^2 - \rho^2$ we can write the quadratic equation in Z as

$$a'Z^2 + b'Z + c' = 0 \tag{3.21}$$

The general solution for Equation 3.21 is given by

$$Z = \frac{-b' \pm \sqrt{(b'^2 - 4a'c')}}{2a'} \tag{3.22}$$

The above Equation 3.22 represents the closed form equation for the value of Z for every point on a spherical object shown in Fig 3.10 and 3.11. We choose only positive and real values of Z.

The results based on our algorithm to generate images of a spherical object with texture as shown in Fig 3.3 are shown in Fig 3.12 and 3.13.

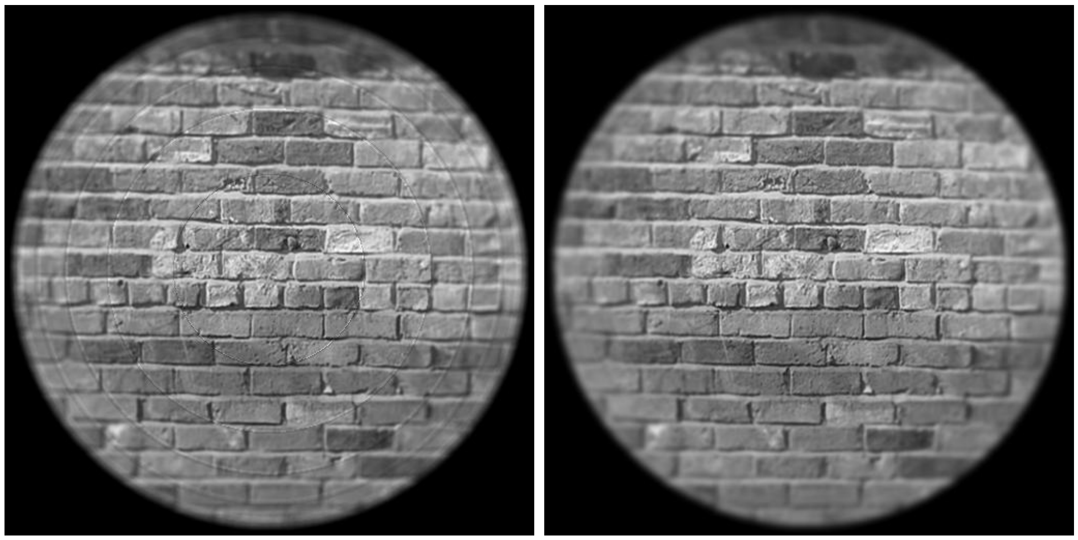


Figure 3.12: Left: Image of spherical object under cylindrical PSF model. Right: Image of cylindrical object under Gaussian PSF model

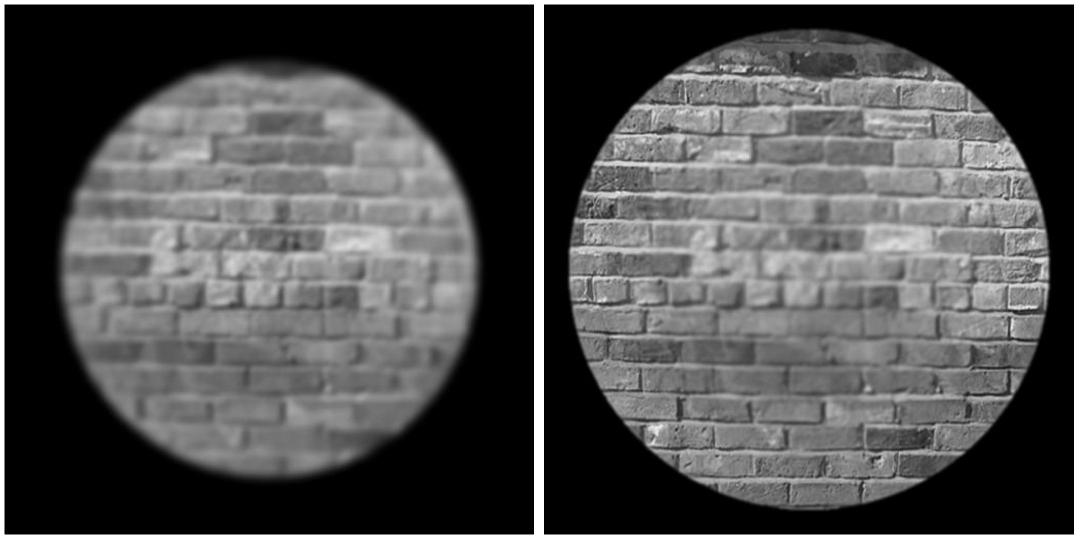


Figure 3.13: Images of spherical object with depth dependent defocus blur

Fig 3.12 left image shows the image of a spherical object using a cylindrical PSF model. We can clearly observe the increase in blur in the radially outward direction i.e. as the object points move away from the lens. The object points on the surface of the sphere near the centre are in focus as they lie in the depth of field region of the imaging system.

Fig 3.12 right image shows the image of a spherical object under the Gaussian PSF model.

Fig 3.13 shows the images of a spherical object with the centre of the sphere placed at varying distances from the lens to illustrate the effect of depth dependent defocus blur.

3.6.4 Conical object

In this section we derive the expression for the distance 'Z' of any point on the outer surface of a right circular cone object whose centre lies on the optical axis as shown in Fig 3.14.

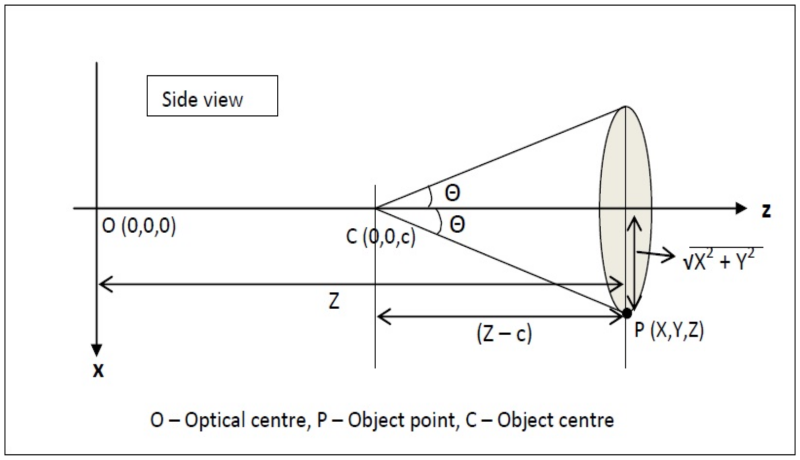


Figure 3.14: Ray diagram illustrating the side view geometry associated in imaging a conical object with apex angle 2θ

The shape specific constant for a cone object is the apex half angle θ and the position of its apex point - C (see Fig 3.14).

We derive the relation between the distance of any point P on the outer surface of the cone from the lens. The distance Z will be expressed in terms of known parameters related to camera and object shape.

From Fig 3.14 we can write

$$tan\theta = \frac{\sqrt{X^2+Y^2}}{Z-c} \tag{3.23}$$

Terms in Equation 3.23 can be rearranged to as follows

$$(Z-c)^2 = \cot^2\theta(X^2+Y^2) \tag{3.24}$$

Using perspective projection formulae we can reduce Equation 3.24 to

$$(Z-c)^2 = \cot^2\theta\left(\left(\frac{xZ}{f}\right)^2+\left(\frac{yZ}{f}\right)^2\right) \tag{3.25}$$

We can reduce the Equation 3.25 to be able to express Z in terms of known parameters as follows:

$$(Z-c)^2 = \cot^2\theta\left(\frac{Z^2}{f^2}(x^2+y^2)\right) \tag{3.26}$$

$$(Z-c)^2 = Z^2\frac{\cot^2\theta}{f^2}(x^2+y^2) \tag{3.27}$$

$$\frac{(Z-c)^2}{Z^2} = \cot^2\theta\left(\frac{(x^2+y^2)}{f^2}\right) \tag{3.28}$$

$$1 - \frac{c}{Z} = \left(\pm\sqrt{x^2 + y^2}\right)\frac{\cot\theta}{f} \tag{3.29}$$

Lets denote $a' = \left(\pm\sqrt{x^2 + y^2}\right)\frac{\cot\theta}{f}$

Thus Equation 3.29 becomes

$$1 - \frac{c}{Z} = \pm a' \tag{3.30}$$

$$1 \pm a' = \frac{c}{Z} \tag{3.31}$$

$$Z = \frac{c}{1 \pm a'} \tag{3.32}$$

The above Equation 3.32 represents the closed form equation for the value of Z for every point on a conical object shown in Fig 3.14. We choose only positive and real values of Z.

The results based on our algorithm to generate images of a conical object with texture as shown in Fig 3.3 is shown in Fig 3.15. The radius of the circular cone is assumed to be greater than the field of view of the imaging system. Thus the image of the conical surface covers the complete image detector area.

Fig 3.15 on the left shows the image of a conical object based on cylindrical PSF model. The object points on the surface of the cone near the centre are in focus as they lie in the depth of field region of the optical system. On the right we have an image correponding to the Gaussian PSF model for the same object.

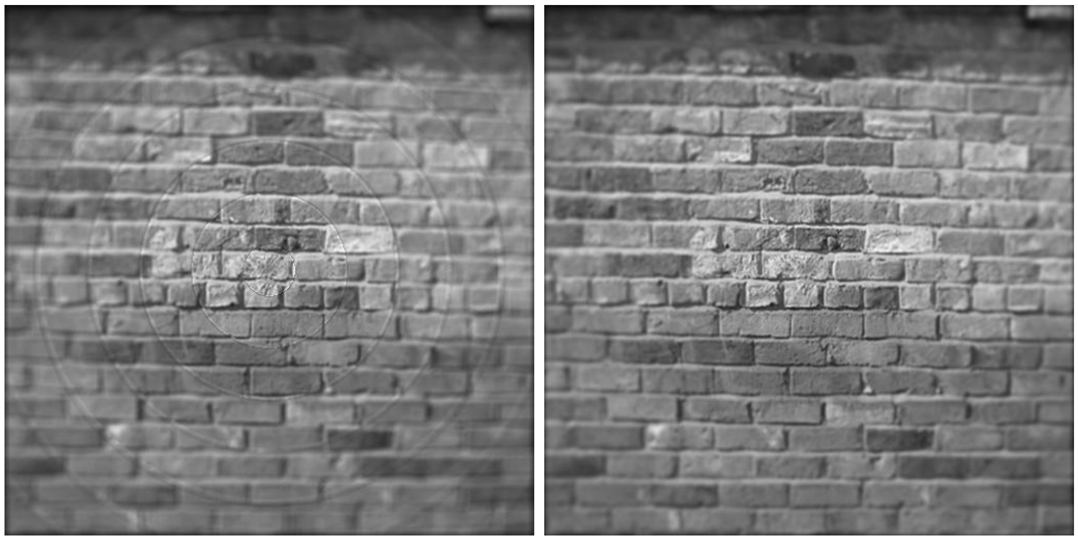


Figure 3.15: Images of conical object with depth dependent defocus blur

3.6.5 Elliptical object

In this section we derive the expression for the distance 'Z' of any point on the outer surface of an ellipsoid whose centre lies on the optical axis as shown in Fig 3.16.

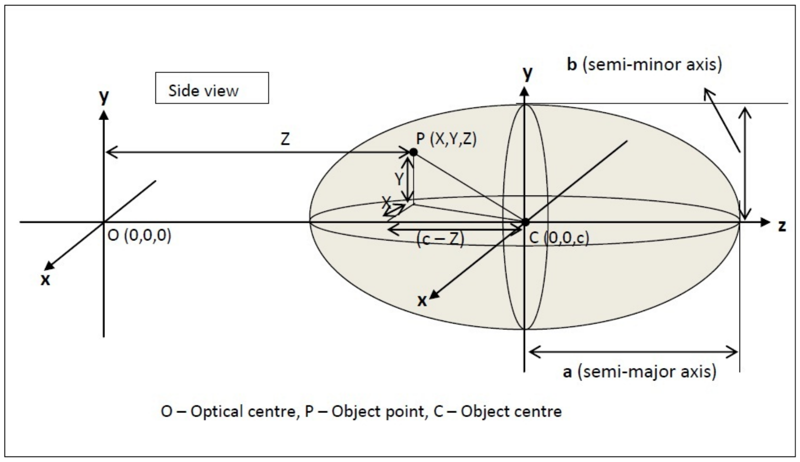


Figure 3.16: Ray diagram illustrating the side view geometry associated in imaging an ellipsoid object

The general equation of an ellipsoid with semi-major axis = a, semi-minor axis

$= b$, semi-vertical axis $= c$ and centre at $(0, 0, 0)$ is given by

$$\frac{X^2}{a^2} + \frac{Y^2}{b^2} + \frac{Z^2}{c^2} = 1 \tag{3.33}$$

The shape specific constants for a ellipsoid object are the semi-major axis $= a$, semi-minor axis $= b$, semi-vertical axis $= c$ and position of its centre 'C' (see Fig 3.14).

In our experiments we consider a simplified version of the ellipsoid with semi-major axis $= a$, semi-minor axis $= b$, semi-vertical axis $= b$ and centre is at $(0, 0, c)$ as shown in Fig 3.16

Thus in our case we can rewrite the Equation 3.33 using the Fig 3.16 as

$$\frac{X^2}{b^2} + \frac{Y^2}{b^2} + \frac{(c - Z)^2}{a^2} = 1 \tag{3.34}$$

Using the perspective projection relations and expanding a few terms we get

$$\frac{x^2 + y^2}{(bf)^2}(Z^2) + \frac{c^2 + Z^2 - 2cZ}{a^2} = 1 \tag{3.35}$$

After simplification and rearranging the terms we get

$$Z^2 \left(b^2 + \frac{(x^2 + y^2)(a^2)}{f^2} \right) - 2cb^2 Z + b^2(c^2 - a^2) = 0; \tag{3.36}$$

Denoting $a' = \left(b^2 + \frac{x^2+y^2}{f^2} \right)$, $b' = -2cb^2$ and $c' = b^2(c^2 - a^2)$ we can write the quadratic equation in Z as

$$a'Z^2 + b'Z + c' = 0 \tag{3.37}$$

The general solution for Equation 3.37 is given by

$$Z = \frac{-b' \pm \sqrt{(b'^2 - 4a'c')}}{2a'} \tag{3.38}$$

We choose only positive and real values of Z.

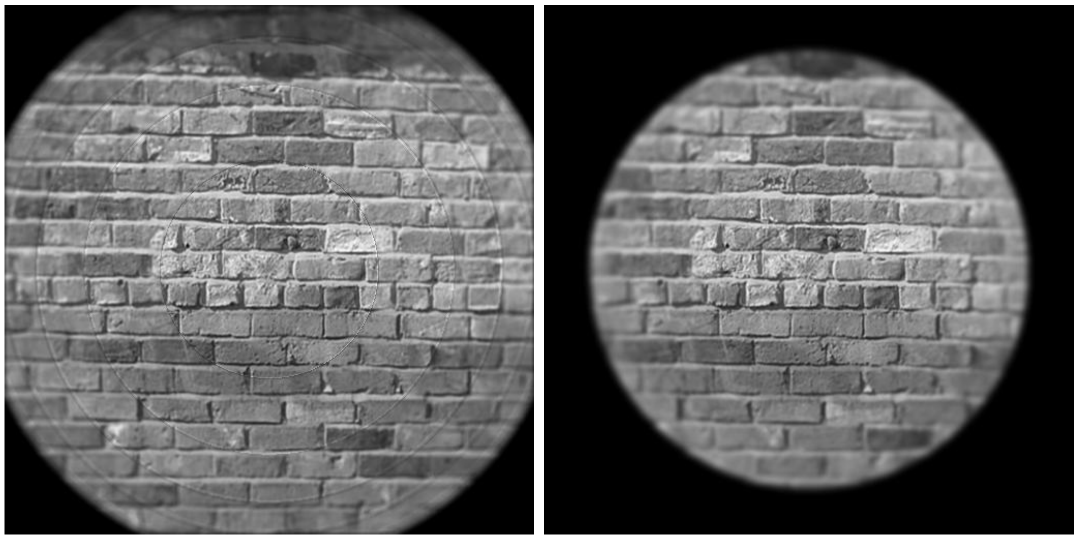


Figure 3.17: Images of conical object with depth dependent defocus blur

The results based on our algorithm to generate images of a ellipsoid object with texture as shown in Fig 3.3 is shown in Fig 3.17.

Fig 3.17 on the left shows the image of a ellipsoid object based on cylindrical PSF model. The object points on the surface of the ellipsoid near the centre are in focus as they lie in the depth of field region of the optical system. On the right we have an image of the ellipsoid object under Gaussian PSF model.

3.6.6 General Quadratic Surface

Through our analysis and derivations in Sections 3.6.1 - 3.6.5 we have addressed the method of depth calculation for points on objects with specific shape in terms of known camera parameters and shape constants. We can generalize our algorithm for any 3D quadratic surface which is expressed as follows

$$a_{200}X^2+a_{020}Y^2+a_{002}Z^2+a_{110}XY+a_{011}YZ+a_{101}ZX+a_{100}X+a_{010}Y+a_{001}Z+a_{000}=0 \tag{3.39}$$

As illustrated in the previous derivations we can use perspective projection formulae and reduce Equation 3.39 to a quadratic equation in Z and solve for real and positive values of Z.

Using the above expressions we can extend the algorithm to support complex objects made up of the primary shapes described in this chapter.

3.7 Summary

Through this chapter we have presented our algorithm and results for generating shift variant defocus blurred images of 3D objects. The camera parameters such as aperture diameter, focal length and the distance of the image detector from the lens are known. These methods and results can be applied to generate highly realistic animation with depth of field effects.

So far our algorithms have been restricted to stationary objects. In the next chapter we introduce the methods to generate shift variant blurred images of moving objects.

Chapter 4

Generating Images of 3D Objects in motion

In the previous chapter we presented an algorithm and results associated with the generation of images of stationary 3D objects simulating the presence of depth dependent defocus blur. In this chapter we extend our algorithm to moving objects.

We consider rigid body motion (translation and rotation) of 3D objects. We make use of equations 2.6 to 2.14 for determining the new position of an object point after translational and rotational motion.

Motion blur is an important visual effect that indicates the motion of an object in an image. Images of 3D objects under motion will not only contain shift variant defocus blur but also contain motion blur. Motion blur is caused due to change in object's position relative to the camera within the exposure time during imaging. The exposure time is understood as the duration for which the shutter is kept open. During this period the image sensor accumulates the incident light energy to result in an image.

4.1 Literature survey

Potmesil et al [11] present a method to simulate motion blur in computer generated images. They model the degradation function due to motion as a shift invariant PSF (SIPSF). The blurred image can then be computed by the time domain convolution based approach or the frequency domain Fourier transform based approach. They have demostrated the results on a scene with 3D objects, each moving in a different path and velocity. Gonzalez et al. [8] provide similar analysis of shift invariant motion PSF. They derive the expression of 2D SIPSF in case of objects moving under uniform linear motion.

4.2 Modeling Motion Blur

4.2.1 Simplified model of the PSF for motion blur

The PSF is modeled as a step function for simplicity. This can be explained as follows. Let the initial position of an image point corresponding to an object point P be $I(a, b)$. During the exposure time of the imaging system the object point P moving at constant velocity results in the displacment of the image point I. Let the final position of the image point be $I(a + \Delta x, b + \Delta y)$ after having travelled a distance of $\sqrt{\Delta x^2 + \Delta y^2}$ (see Fig 4.1). We model the blurring process by equally distributing the intensity of the image point over all the pixels in the path from $I(a, b)$ to $I(a + \Delta x, b + \Delta y)$ in the direction of motion. This model can be represented as a

2D PSF as follows

$$h(x,y) = \begin{cases} \frac{1}{\sqrt{\Delta x^2+\Delta y^2}} & a \leq x \leq (a+\Delta x), b \leq y \leq (b+\Delta y)\text{: } \textit{in direction of motion} \\ 0 & \textit{otherwise} \end{cases} \tag{4.1}$$

which is in agreement with the property of a lossless imaging systems i.e.

$$\int\int h(x,y)\,dxdy = 1 \tag{4.2}$$

Fig 4.1 shows an example of the motion blur model that we adopt for our simulation.

4.2.1.1 Limitations of the PSF model to represent motion blur

Our model does not account for PSF with respect to non-uniform or accelerated motion. In case of non-uniform motion where the object velocity varies, the coefficients of the PSF will not be equal. The coefficients of the motion blur PSF will have higher weight at pixels where the object's velocity has been lower. This effect is not realized by our model.

4.2.1.2 Software implementation of motion blur model

Our model has been implemented in software similar to multiple exposure photography and CEMENT (Computer Enhanced Multiple Exposure Numerical

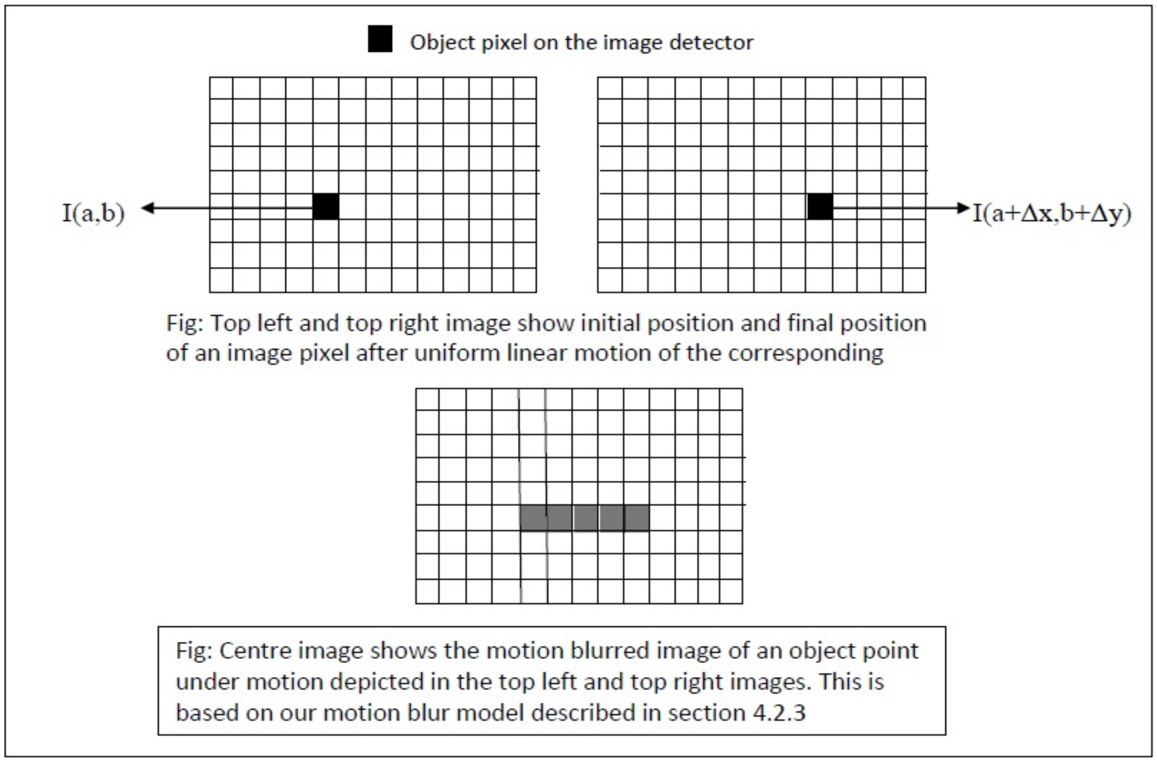

Object pixel on the image detector
I(a,b)
I(a+Δx,b+Δy)
Fig: Top left and top right image show initial position and final position of an image pixel after uniform linear motion of the corresponding
Fig: Centre image shows the motion blurred image of an object point under motion depicted in the top left and top right images. This is based on our motion blur model described in section 4.2.3

Figure 4.1: Illustration of motion blur model used in our experiments. The intensity of the moving object has been equally distributed over the line of displacement

Technique). We compute a set of frames which are at equally sampled time instants of the total exposure time and average them to determine the motion blurred image. A sample frame computed by our algorithm reflects the object's position at that time instance which is calculated based on the the rigid body motion equations (Equation 2.6 to 2.14). An example of such image is shown in Fig 4.2 and 4.3.

4.2.2 Classification of motion blur

Motion blur can be broadly classified into two types based on the motion of objects in the scene as listed below:

1. Shift invariant motion blur

2. Shift variant motion blur

4.2.2.1 Shift invariant motion blur

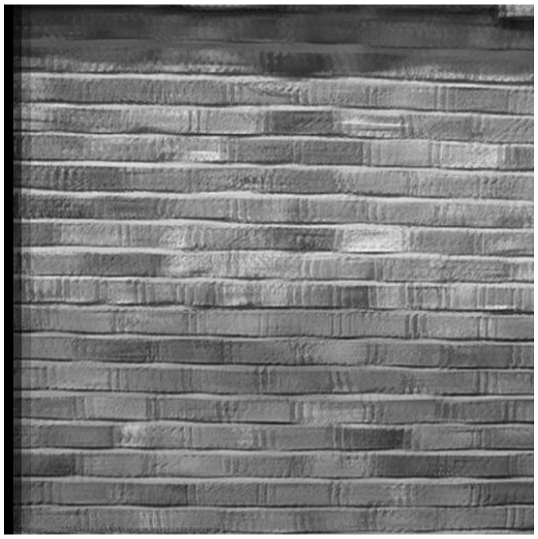


Figure 4.2: Image of a planar object moving in the x direction with constant velocity

The blur caused due to the uniform linear motion of object points such that all their corresponding image points move exactly the same distance. This can happen under the condition that all moving object points are at the same depth or at the same distance from the lens. For example, a planar object placed parallel to the lens moving in x and/or y direction only, with constant velocity but without change in its depth. The image of such an object will contain shift invariant motion blur. Under shift invariant motion blur it is needless to say that the PSF at all points on the image plane due to motion is the same.

Fig 4.2 shows an image of a planar object moving with a constant velocity in the x direction and containing shift invariant motion blur. Details of how we generated this image will be given in section 4.3

4.2.2.2 Shift variant motion blur

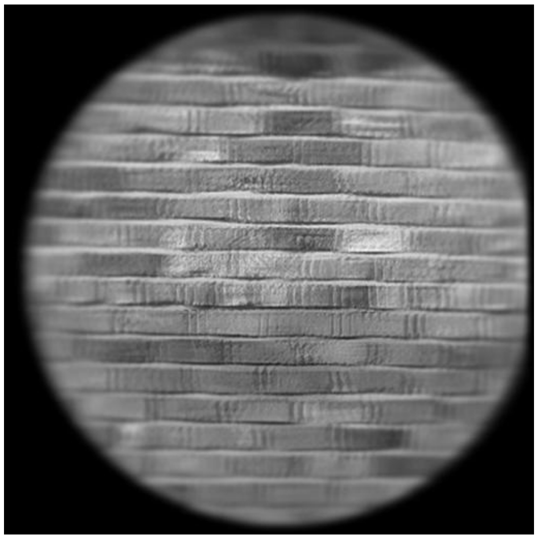


Figure 4.3: Image of a spherical object moving in the x direction with constant velocity

When the blur at every point on the image due to motion of the object points varies we can say that the image will contain shift variant motion blur. This can be caused due to the following reasons:

1. Different object points move at different velocities in the scene.

2. Magnification effects of all optical systems. This effect is largely seen due to the varying depth characteristics of scene points.

Due to the magnification effects, objects points at different depths in the scene appear to travel varying distances on the image detector inspite of experiencing uniform linear motion. Thus the PSF at each point on the image plane varies and is dependent on the distance and direction of travel of object points.

Fig 4.3 shows an image of a spherical object moving with a constant velocity in the x direction. It contains shift variant motion blur. Details of how we generated this image is given in section 4.3

4.3 Algorithm to generate images of moving objects

Our algorithm computes the image of 3D objects under motion. These images will include the effects of defocus blur and motion blur based on the shape of the object and motion parameters. Our software SCO-3D generates synthetic images which are close to the images produced by a real digital camera with limited depth of field. The software is capable of generating images equivalent to frames captured by a video camera used to shoot moving objects. As an example, the image of moving sphere can be seen in Fig 4.3 which contains shift variant defocus blur and shift variant motion blur.

In this section we explain the assumptions, input parameters and algorithm to generate images of 3D objects under motion. The motion can be translational and/or rotational in 3 dimensional space.

4.3.1 Assumptions

- The opening and closing of the shutter is considered to be instanteneous and thus we do not need to take into account degradation due to shutter PSF.

- The optical medium does not introduce any degradation effect on the light passing through it.
- All shading and shadow effects in the image of the object are ignored.
- Our PSF model for motion blur does not account for non-uniform motion of the object points.
- The image is considered to be smooth in a 3x3 neighborhood around any pixel.

4.3.2 Input parameters

Along with the input parameters listed in 3.3.2 and their default values we will need the following input parameters to specify object motion.

- v$_x$, v$_y$, v$_z$ - The velocity of the object in x, y and z directions respectively (Unit: mm/sec).
- ω_x, ω_y, ω_z - The angular velocity of object points about the x, y and z axis of the object coordinate system (Unit: deg/sec).
- t - The time interval between 2 consecutive frames of a video that captures moving objects (Unit: secs).
- n - Number of video frames to be genarated

4.3.3 Computation of blurred image with motion blur and defocus blur

We sample the interval between 2 frames into equally spaced time instants and generate an intermediate frame at each time instant. This intermediate frame not only captures the motion of individual object pixels (using Equation 2.6 to 2.14) but also contains the respective amount of defocus blur based on object shape. The number of intermediate frames has been chosen to be 5 for our experiments. This number is based on the fact that modern digital cameras support multiple exposure photography with 3 to 10 shots taken in succession to produce one final image. This number can be changed based on the constraints posed by the optical system. The intermediate frames are then averaged to give the image of the moving object with motion blur. It is worth noting that the image points corresponding to object points nearer to the lens undergo farther displacement than the ones away from the lens. Magnification effects governed by the perspective projection relations demonstrate this phenomenon. For ex. let us consider 2 object points, one at a depth of 250 mm from the lens and the other at a depth of 285 mm from the lens. If these points move by a distance of 10 mm in the x direction, then the displacements of their image points will be 80 pixels and 70 pixels respectively given that the image detector is placed at the focal point (which is at a distance of 20 mm from the lens).

The computation of intermediate frames is performed in the following manner:

1. Store the intensity and depth information (Z value) of all object pixels calculated based on steps 1 to 3 of section 3.3.3 described in Chapter 3.

2. Using the rigid body motion formulae (Equation 2.6 to 2.14) and the values of

the motion parameters provided as input, compute the new position (in 3D) of each object point.

3. Based on the new position determine its respective image plane position using the perspective projection relations.

4. If the image pixel is within the scope of the image detector then re-calculate the PSF due to defocus blur with its new depth value and generate the image with defocus blur as described in steps 4 to 6 of section 3.3.3 described in Chapter refchap:2.

5. Finally, all intermediate images between 2 frames are averaged to produce a video frame with shift variant defocus blur and shift variant motion blur.

We maintain a matrix of values called Validity map for every intermediate frame which indicates if a pixel has a valid intensity and depth value. Background pixels will have complex values for depth and a set of other pixels will have unknown intensity and depth values. These pixels with unknown values are termed as " Missing pixels" and the reason of their occurence is explained in detail through Figures 4.4, 4.5, 4.6 and section 4.3.4.

We show 3 sample video frames (along with intermediate frames) (see Fig 4.4, 4.5 and 4.6) generated by our software (SCO-3D) of a rotating sphere to demonstrate the effects of defocus blur and motion blur on moving objects. In the intermediate frames and the motion blurred frames we notice regions of low intensity (dark) pixels. This problem is referred as the Missing Pixel problem which is explained in detail in the next section. The set of all missing pixels in all the images (Fig 4.4, 4.5 and 4.6)

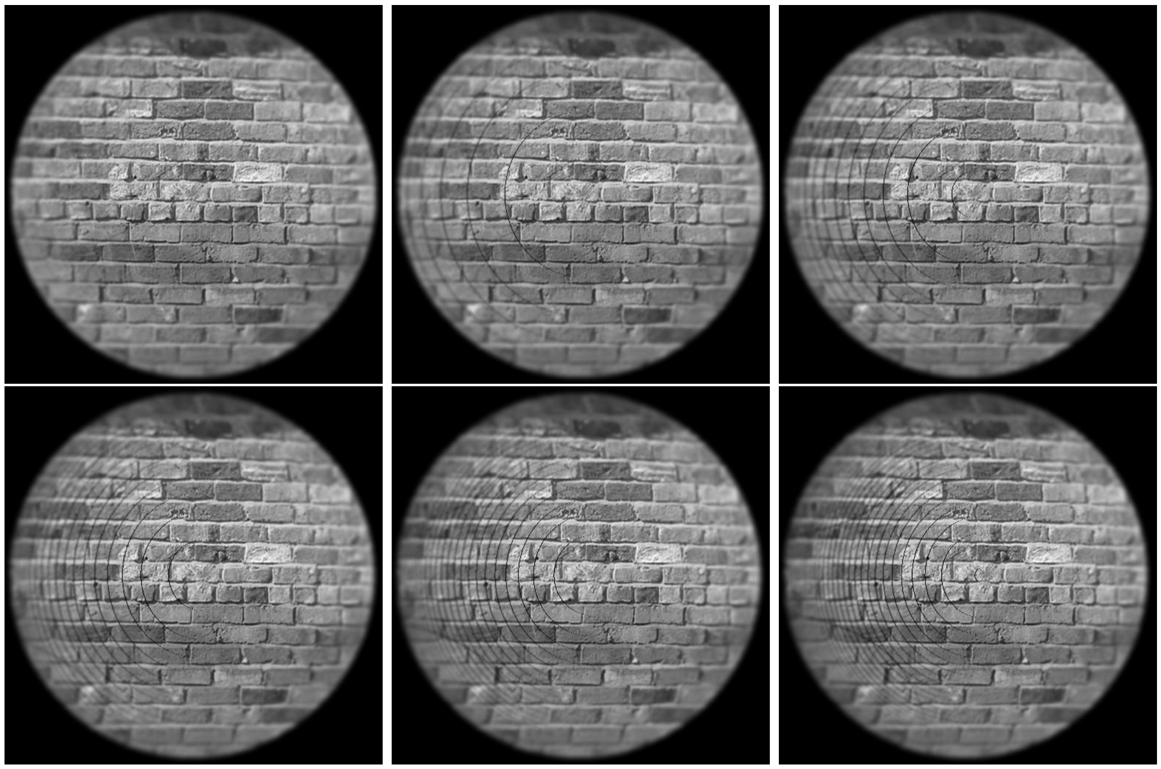


Figure 4.4: Intermediate frames (1 to 6) of a rotating sphere (about the y axis with angular velocity $\omega = 5$ deg/sec) and sampled at every $\frac{1}{5}$ sec or 1° of rotation

has been cumulatively presented (white colored pixels) in Fig 4.7 which is also called as Mask image.

4.3.4 Problem of missing pixels

In images presented in Fig 4.4 and 4.5 we observe different patterns of missing pixels in each intermediate frame and motion blurred frame. This is the result of non-uniform movement of image points on the image detector resulting because of magnification effects of the lens. As we have previously stated that even though object points (at different depths) experience uniform motion (linear or non-linear),

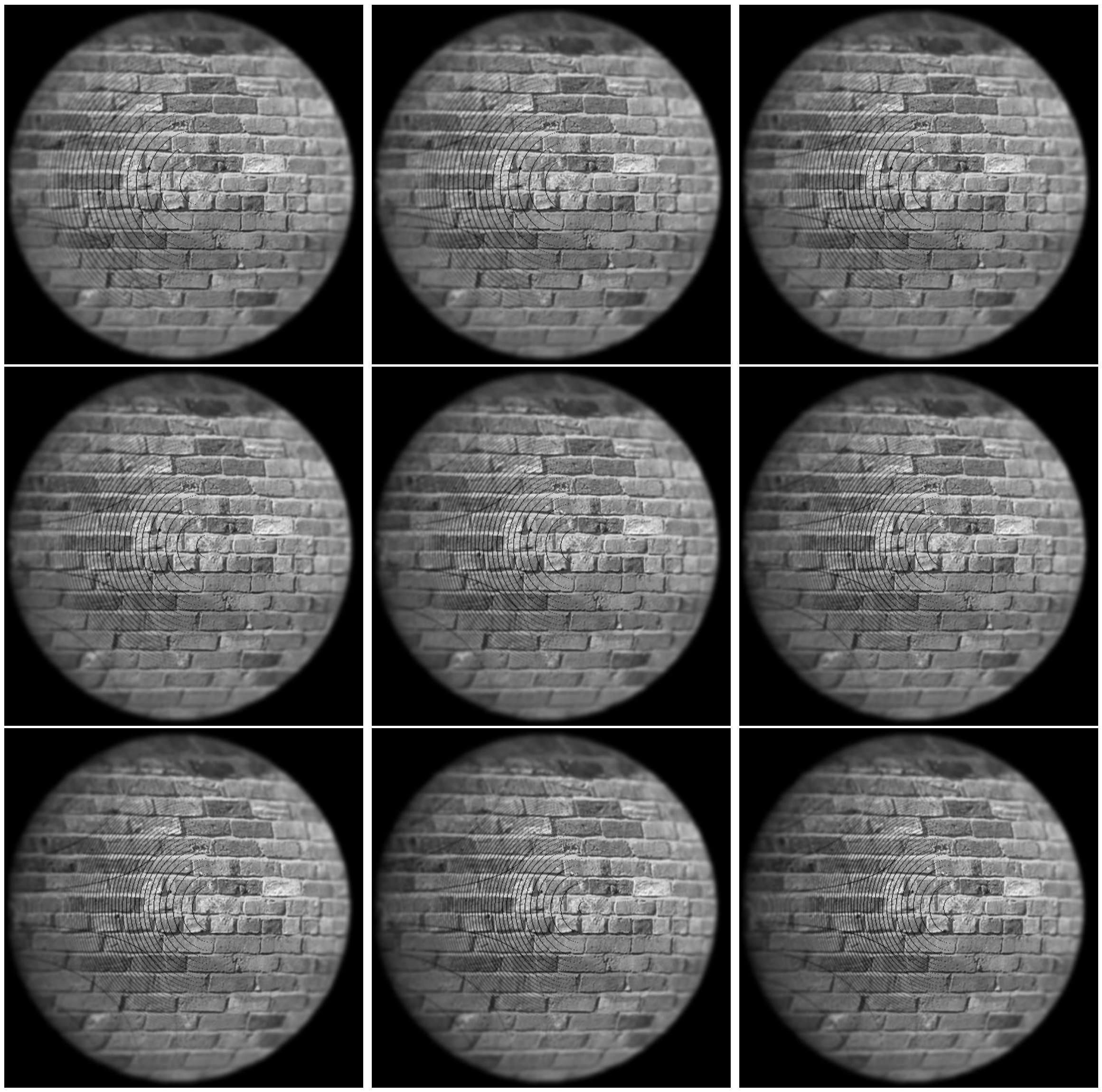


Figure 4.5: Intermediate frames (7 to 15) of a rotating sphere (about the y axis with angular velocity $\omega = 5$ deg/sec) and sampled at every $\frac{1}{5}$ sec or 1° of rotation

they undergo dissimilar displacement on the image detector due their varying depth characteristics. Such effects lead to the phenomenon of missing pixels. For our context we define the problem of missing pixels as "The non-uniform displacement of image points corresponding to the uniform motion of 3D object points with different depths

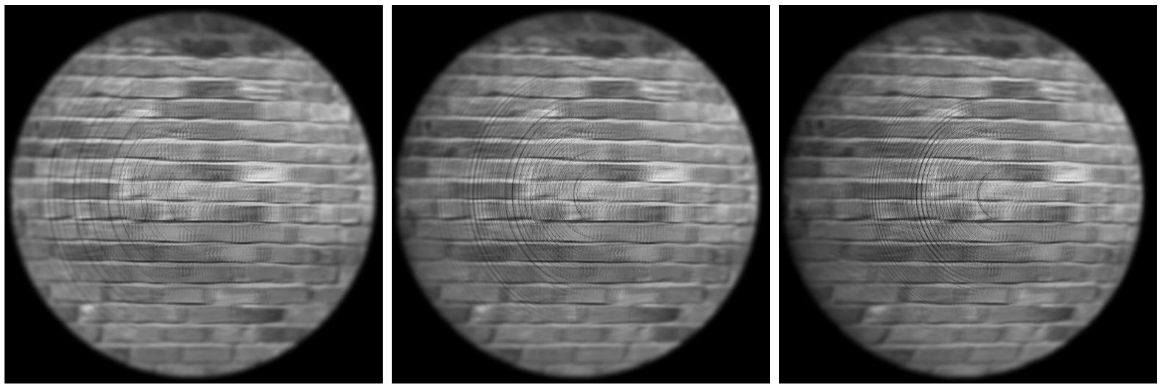


Figure 4.6: Video frames of a rotating sphere (about the y axis with angular velocity $\omega = 5$ deg/sec) captured every 1 sec containing motion blur and defocus blur

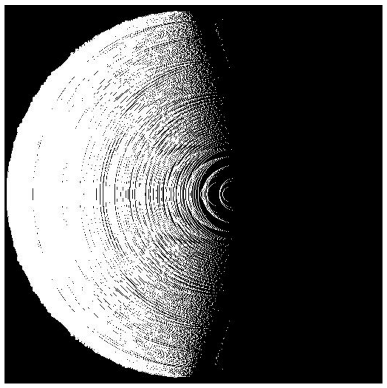


Figure 4.7: Cumulatively presented missing pixels from Fig 4.4, 4.5 and 4.6

resulting in regions of missing or unknown intensity pixels". Thus two object points of different depths that get mapped to neighboring pixels on the image may not get mapped as neighbors after the object points undergo uniform motion. Even in case of non-uniform motion i.e. different object points moving at different velocities such phenomenon is noticed.

We provide examples of such images with missing pixels generated by our software in several cases of moving objects (see Fig 4.8 to 4.12).

Through our experiments we have observed that these regions of missing pixels

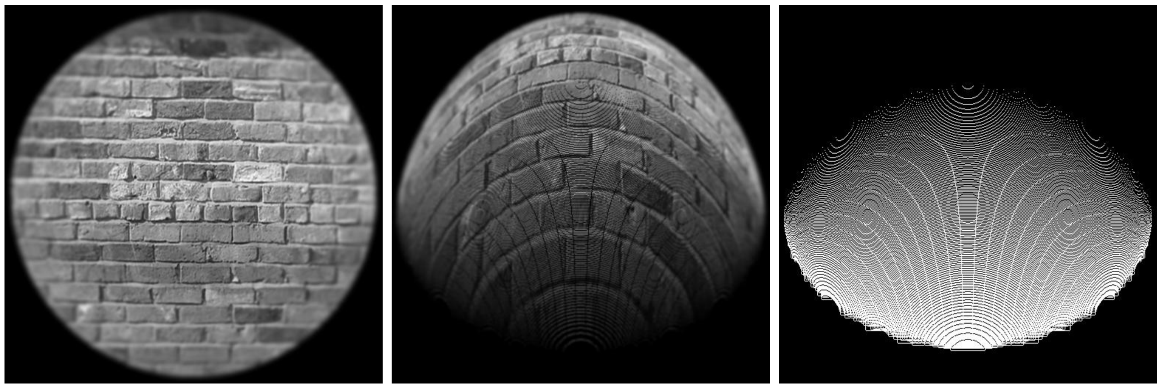


Figure 4.8: Left: Image of a spherical object with shift variant defocus blur. Centre: Image of a spherical object after a 45° rotation about the x axis with missing pixels. Right: Mask image corresponding to missing pixels

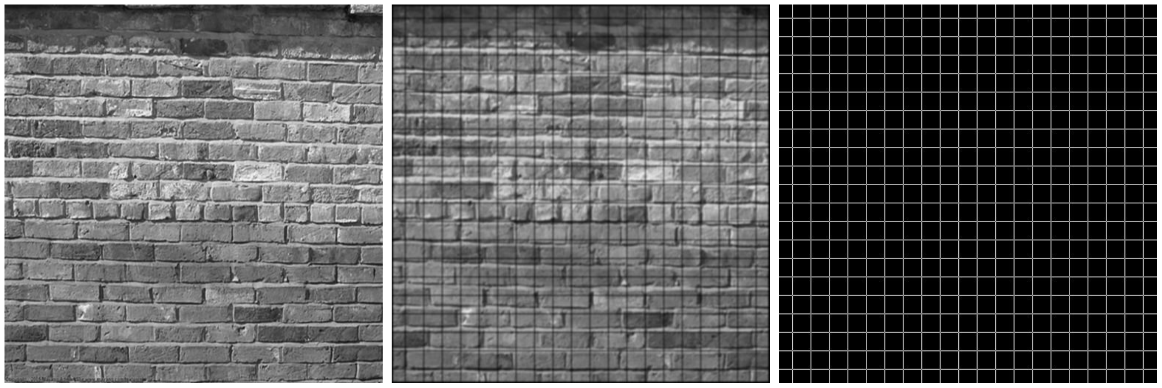


Figure 4.9: Left: Image of a planar object in focus. Centre: Image of a planar object after a moving a distance of 10 mm away from the lens with missing pixels. Right: Mask image corresponding to missing pixels

display specific patterns and the missing pixels are not concentrated in a particular region. In our experiments we have a single object under translational and/or rotational motion. We observe that the missing pixels are surrounded by a set of so called known (non-missing) pixels. Known pixels are the image points for which the intensity and depth value of their corresponding object points (after motion) are known. These known pixels can be useful in filling the missing pixels by using interpolation techniques. In the next section we provide a heuristic solution to the problem of missing pixels with analysis and results.

4.3.5 A heuristic algorithm for filling missing pixels based on interpolation techniques

As per our observations the missing pixel patterns display certain properties which can be exploited for developing algorithms to resolve them. We have seen that missing pixels are not clogged in one area of the image to create a big hole or a randomly shaped region of missing pixels. The patterns of missing pixels are such that for any missing pixel there are a few known neighbor pixels (maximum upto 8 in a 3x3 neighborhood). The information of these known neighbors can be used to fill in an appropriate value into the missing pixel. The missing pixel not only represents a missing intensity value but also represents a missing depth value (Z value). Naive linear interpolation methods have been observed to perform

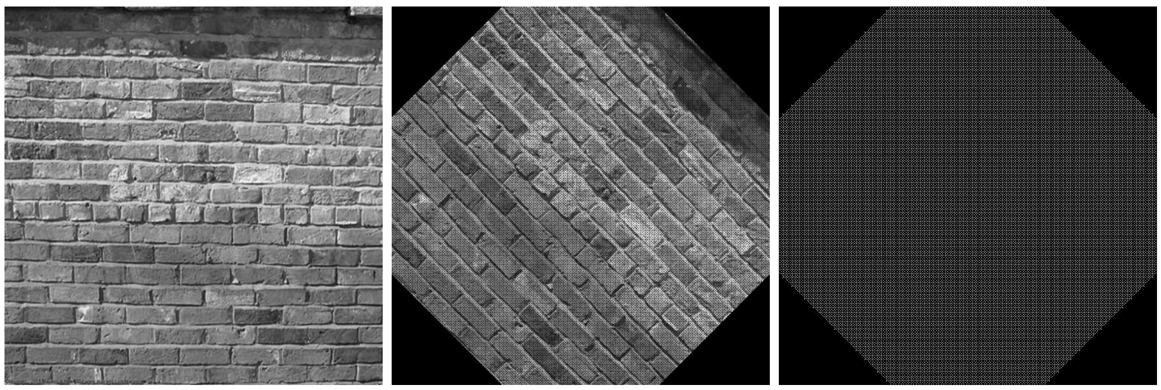


Figure 4.10: Left: Image of a planar object in focus. Centre: Image of a planar object after a 45° rotation about the z axis with missing pixels. Right: Mask image corresponding to missing pixels

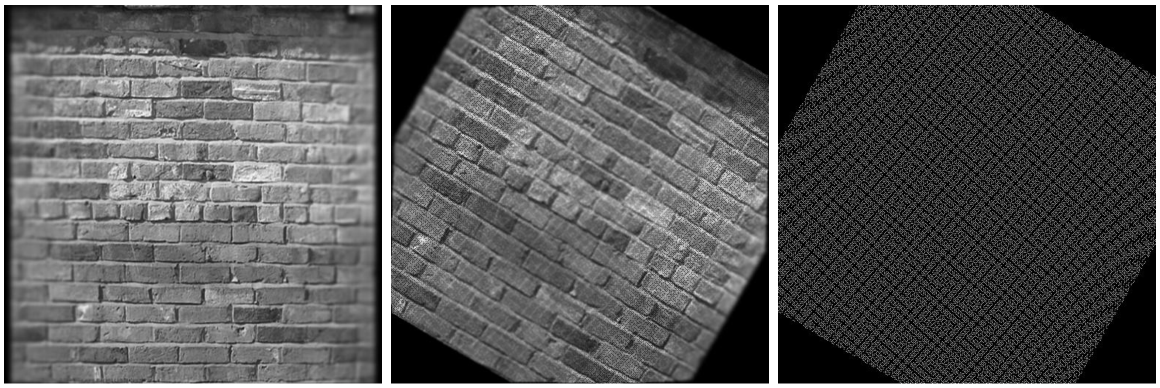


Figure 4.11: Left: Image of a cylindrical object with shift variant defocus blur. Centre: Image of a cylindrical object after a 30° rotation about the z axis and 10 mm translational in the z direction away from the lens showing missing pixels. Right: Mask image corresponding to missing pixels

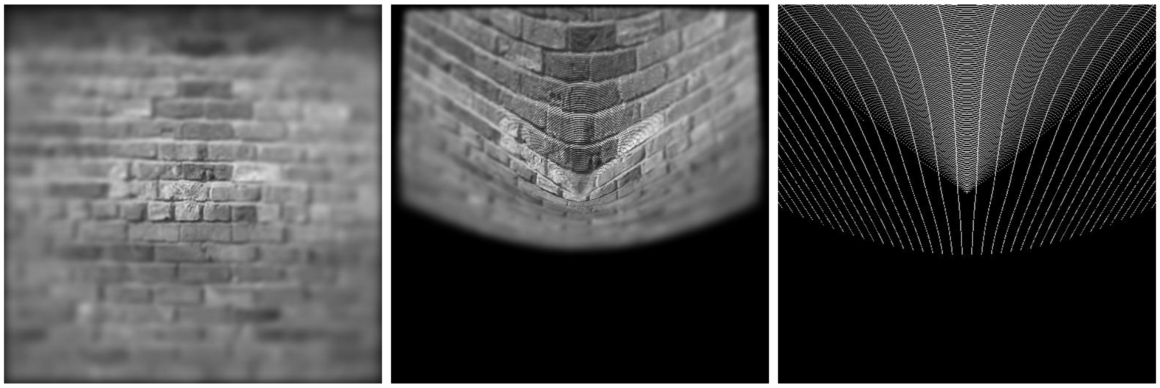


Figure 4.12: Left: Image of a conical object with shift variant defocus blur. Centre: Image of a conical object after a 20° rotation about the x axis and 10 mm translation in the z direction away from the lens showing missing pixels. Right: Mask image corresponding to missing pixels

ineffectively. The distances of nearest known pixels from the missing pixel can vary in many cases. For example, in Fig 4.9 we see that the pattern of missing pixels is seen as straight lines. Thus, given a pixel at the intersection of 2 lines we may not have any known neighbor pixels in either x direction or y direction. Thus any linear interpolation technnique based on 4 neighbor approach will fail in this context.

However, we observe that there are known diagonal pixels in the 3x3 neighborhood which can be used for interpolating a value to the centre missing pixel. We have developed an algorithm based on nearest known neighbor pixel of a missing pixel, preferably in a 3x3 neighborhood and fill it using Adaptive Interpolation Technique (AIT). The algorithm is supported by the assumption that the image is considered to be smooth in a 3x3 neighborhood.

Our algorithm is split into 2 parts with the first part focused on detection of missing pixels and the second part based on assigning appropriate values to (filling up) these missing pixels.

4.3.5.1 Detection of missing pixels

Before employing interpolation techniques to fill the missing pixels we need an algorithm to efficiently detect the missing pixels. We use a technique based on row-wise and column-wise scanning to detect missing pixels. The steps involved can be listed as follows:

1. Using the validity map constructed through our algorithm (see section 4.3.3) perform a row-wise scan of the the intermediate frame and determine a set of end point pixels (left end point and right end point) which have known values of intensity and depth.

2. If the left end point and right end point pixels are the same or if there are no end point pixels detected then we will attempt to detect and resolve pixels in the column wise scan.

3. If the end point pixels detected have distinct positions then we search for missing

pixels between the end point pixels of the row under consideration by checking for their validity. These missing pixels will be immediately filled using the algorithm described in section 4.3.5.2. Pixels that are filled are marked as valid but interpolated.

4. The above steps are repeated during a column-wise scan so as be able to spot pixels that were undetected during row-wise scan. These missing pixels will be immediately filled using the algorithm described in section 4.3.5.2. Pixels that are filled are marked as valid but interpolated.

Pixels marked as valid but interpolated are not used for interpolating values into missing pixels until one iteration (one row wise scan and one column wise scan) of the algorithm is performed. This is done so as to reduce the propagation of error in subsequent frames.

4.3.5.2 Algorithm to resolve the problem of missing pixels based on adaptive interpolation techniques (AIT)

We will use an adaptive combination of 3 simple methods in a specific order of preference to compute the intensity and depth value for all missing pixels. A particular order is followed so as to ensure faster computation.

1. Interpolation using least distant neighbors (3x3 neighborhood) - The 3x3 neighborhood of the missing pixel is scanned to determine two known pixel values. Both of these known pixels will be selected from either the horizontal direction, vertical direction or any or the diagonal directions in the respective order as mentioned. The centre pixel which is the missing pixel, is filled via

linear interpolation of the two selected known pixels.

2. Neighbor replication - If we do not find such pairs of neighbors we will copy the first found nearest neighbor in a 3x3 neighborhood of the missing pixel.

3. Boundary processing - Missing pixels along the border of the image i.e. the first row, first column, last row and last column will not have all the eight neighbors (3x3 neighborhood). Thus for such pixels we use the linear interpolation of nearest known pixels in either horizontal (for row-wise scan) or vertical direction (for column-wise scan).

The above algorithm has been tested over several images of different object shapes that were mentioned in section 3.1. Our experiments have shown that the solution based on adaptive interpolation techniques (AIT) provides good results for our missing pixel problem. It has also been observed that in most cases the algorithm resolves all missing pixels with just one row-wise and one column-wise scan of the image. However, in some rare cases another interation of AIT may be required. This is handled adaptively by tracking the number of missing pixels and number of pixels resolved during the current iteration.

Our software SCO-3D incorporates the above mentioned algorithm. The results of our algorithm on images shown in Figures 4.8 to 4.12 can be seen in Figures 4.13 to 4.17.

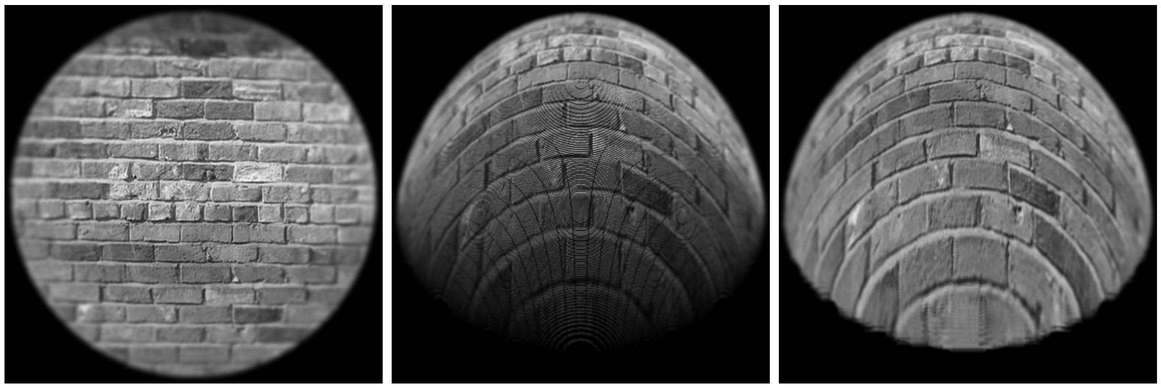


Figure 4.13: Left:Image of a spherical object with shift variant defocus blur. Centre, Right: Image of a spherical object after a 45° rotation about the x axis with missing pixels and after processing using AIT. Number of missing pixels resolved - 68804

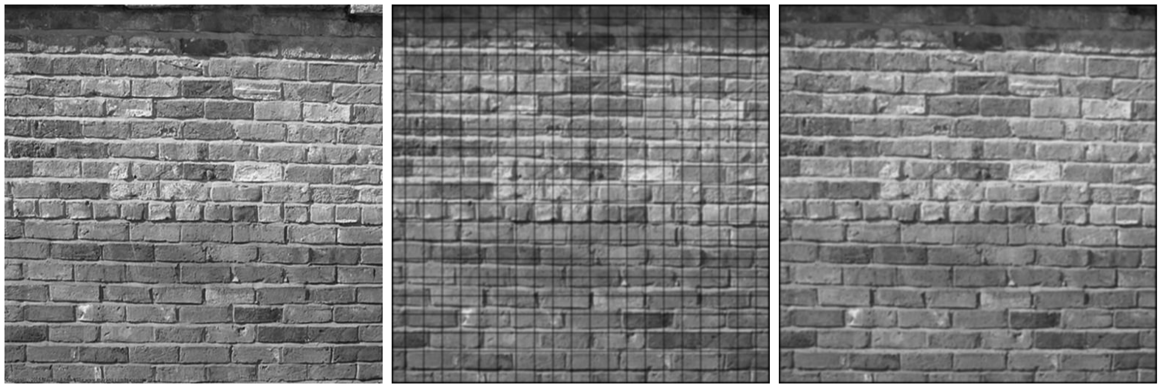


Figure 4.14: Left: Image of a planar object in focus. Centre, Right: Image of a planar object after a moving a distance of 10 mm away from the lens with missing pixels and after processing using AIT. Number of missing pixels resolved - 19680

4.3.6 Other challenges faced during the implementation of our algorithm

4.3.6.1 Occlusion detection and solution

During the generation of such images we efficiently take care of occlusion effects and conditions under which any part of the object is not within the scope of the

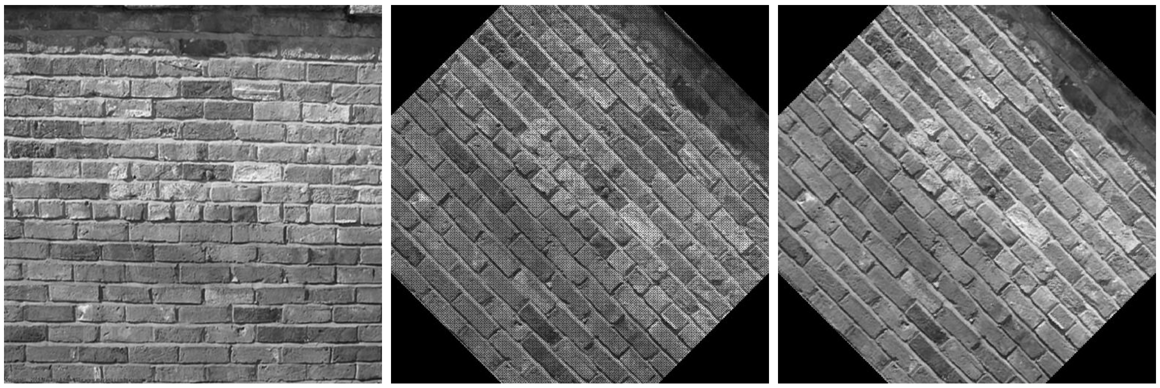


Figure 4.15: Left: Image of a planar object in focus. Centre, Right: Image of a planar object after a 45° rotation about the z axis with missing pixels and after processing using AIT. Number of missing pixels resolved - 36993

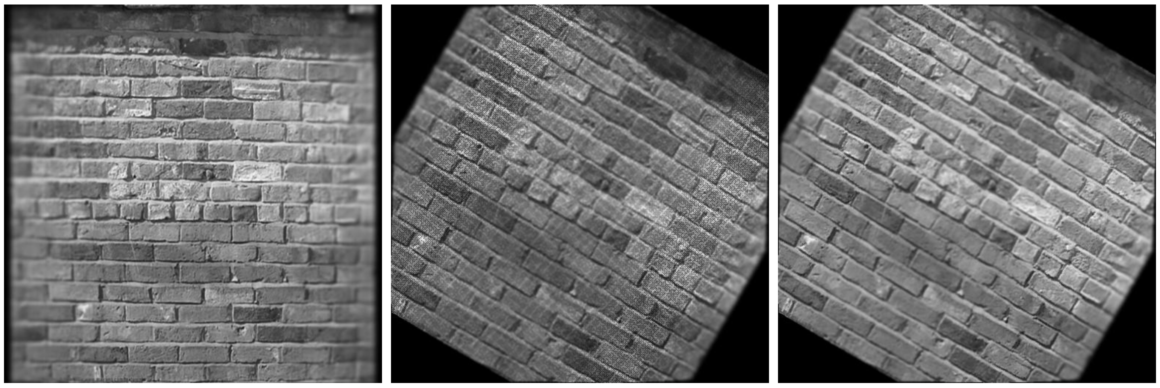


Figure 4.16: Left: Image of a cylindrical object with shift variant defocus blur. Centre, Right: Image of a cylindrical object after a 30° rotation about the z axis and 10 mm translational in the z direction away from the lens with missing pixels and after processing using AIT. Number of missing pixels resolved - 36784

camera. The occlusion effects are seen due to the mapping of two or more object points to the same pixel on the image. This happens due to non-uniform motion of corresponding image points of moving 3D objects points. We resolve the occlusion problem by considering the pixel closest to the lens when compared to overlapping pixels farther away from the lens. Parts of the object which are out of scope of the camera are ignored during processing.

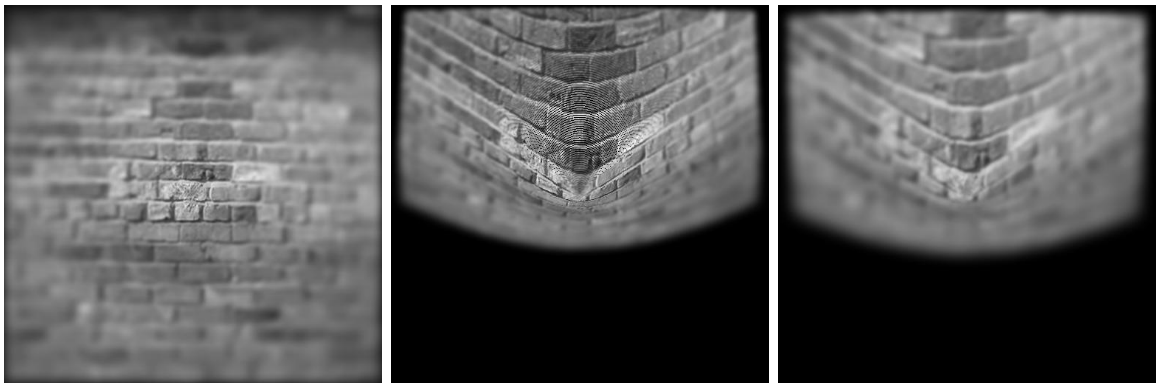


Figure 4.17: Left: Image of a conical object with shift variant defocus blur. Centre, Right: Image of a conical object after a 20° rotation about the x axis and 10 mm translation in the z direction away from the lens with missing pixels and after processing using AIT. Number of missing pixels resolved - 26122

4.3.6.2 Generating multiple frames from a given test image

We use the initial blurred image (with defocus blur only) of the stationary object to determine and generate all the rest of the intermediate frames simulating object motion. We do not use the previous frame information to generate the next frame so as to avoid the propagation of noise which gets cumulatively added. Noise in our context is defined as round-off errors and errors introduced due to interpolation during the generation of subsequent video frames. Thus, we will always use the original blurred image of the stationary object (with intensity map and depth map) to generate the next set of video frames.

4.3.7 Limitations of Adaptive Interpolation Technique (AIT)

The AIT algorithm has been observed to provide good results in resolving the problem of missing pixels in intermediate frames in our experiments. The intermediate frames were generated by our algorithm (see section 4.3.3) for the simulation of defocus blur and motion blur present in images of 3D moving objects. The patterns exhibitied by missing pixels display a characteristic of not having large random regions. This characteristic is exploited by AIT. However, in cases of large regions of randomly distributed missing pixels the performance of AIT may not be guaranteed. The steps in the algorithm (see section 4.3.5) display an inherent direction bias (initially in the horizontal direction) during interpolation. This algorithm may not be utilizing useful information of other neighboring pixels which are at the same distance away from the missing pixels as the two pixels selected for interpolation. Thus, we would need a more generic algorithm useful for filling missing pixels that are randomly distributed over large regions of the image. Several instances where such images with large missing pixel regions occur, have been quoted in [12, 13, 14, 15, 16, 17, 18, 19, 20, 21]. The problem is also referred to as Image Inpainting with reference to work published by [17, 16, 14, 12, 21]. We present a description of current methods, our novel algorithm and a qualitative analysis to address this problem in the next chapter.

Finally we present a set of results generated by the help SCO-3D. The image in Fig 4.19 is equivalent to a video frame captured by a video camera mounted on a moving car. The camera optical axis makes an angle of $30°$ to the road. The images in Fig 4.18 represent the top view and intermediate frames constructed by our software to generate the motion blurred frame as shown in Fig 4.19.

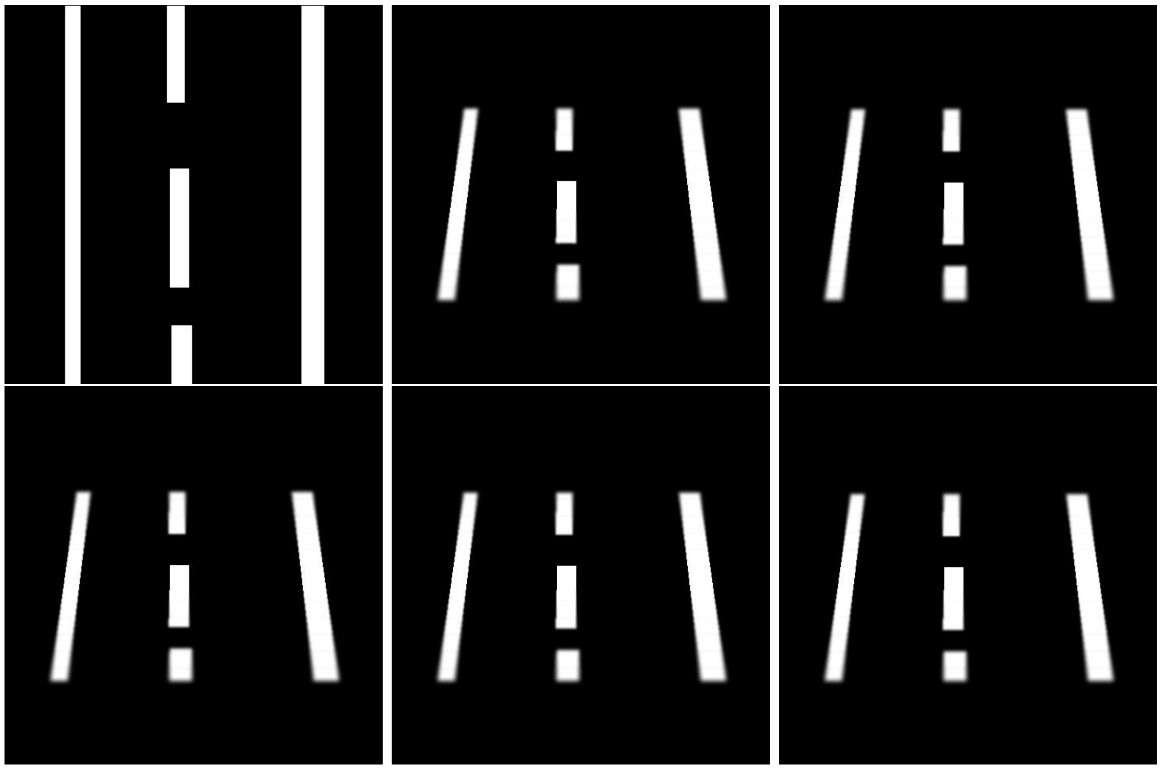


Figure 4.18: Top view of road and intermediate frames (1 to 5) as would be captured by a video camera mounted on top of a moving car.

4.4 Summary

In this chapter we have presented an algorithm to generate images of moving 3D objects. These images will contain both shift variant defocus blur and shift variant motion blur. We also resolve the problem of missing pixels which occurs during the generation of such images using an adaptive interpolation technique. These methods and results can be applied to generate highly realistic animation with depth of field and motion blur effects.

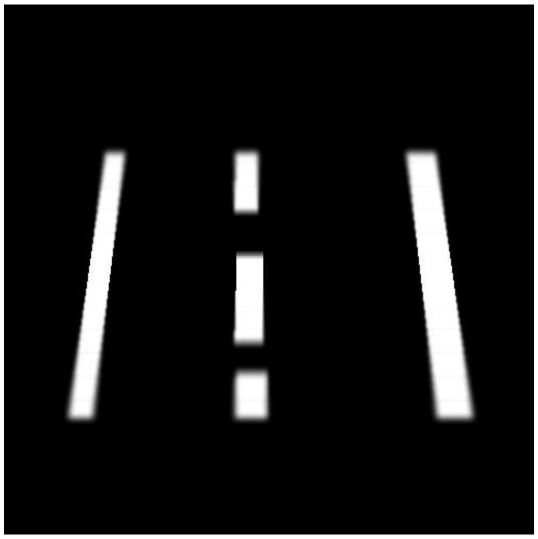


Figure 4.19: Image representing a single frame as would captured by a video camera mounted on a moving car.

Chapter 5

Generic Algorithm for the Problem of Missing Pixels

In the previous chapter we presented our algorithm and results to generate images of 3D objects under motion. We also described the problem of missing pixels and proposed the Adaptive Interpolation Technique (AIT) to resolve it. The AIT technique exploited the patterned nature of missing pixels that occured in synthetic images generated for moving 3D objects. In cases of randomly distributed regions with arbitrary size and shape we found that the AIT does not provide satisfactory results. This was because of the direction bias of the algorithm (initially during horizontal direction). AIT does not provide a rotation invariant solution to the problem of missing pixels in general. In this chapter we present a novel approach to resolve the problem of missing pixels independent of a pattern that may be associated with the occurence of missing pixels.

5.1 Problem of missing pixels and image inpainting

The missing pixel problem has been encountered in different applications of image and video processing. In some papers [17, 16, 14, 12, 21] this problem has been referred to as Image Inpainting. It is also referred as hole filling in images.

Some examples where the missing pixel (or image inpainting) problem arises and an efficient solution is necessary are:

- Restoration of ancient pieces of art in museums, which were degraded over time due to their prolonged exposure to the atmosphere.
- Removal of objects from an image in a manner such that it is imperceptible to the viewer. For example, removal of people from Google street view images.
- Transmission errors (holes) that occur during the transfer of streaming image/video data over wired or wireless networks.
- Missing pixels may occur in archived motion picture films or in image sequence from high speed camera [20]. Missing pixels are spotted as regions of random intensity due to physical damage or particles caught in the film during transportation.

Fig 5.1 shows a tampered image with ramdonly distributed regions of missing pixels.

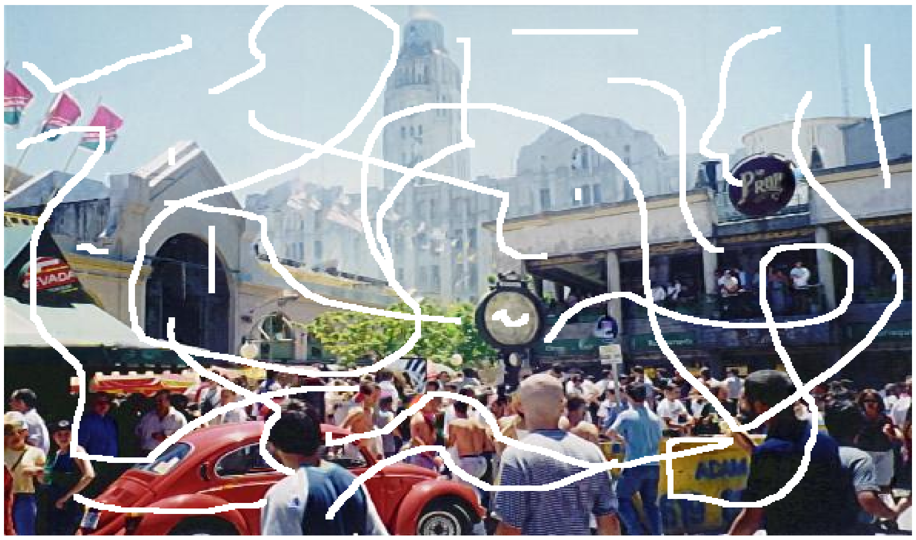


Figure 5.1: Tampered image of a crowd with randomly scattered missing pixels (white colored pixels)

5.2 Literature survey

Kokaram et al. [20] address the missing pixel problem in the context of archived motion picture film. They provide a solution based on markov random field model and 3D autoregressive model of an image. Their methods involve multiresolution block matching utilizing the spatial and temporal features of a video frame. They also employ multilevel median filter for noise suppresion.

Efros et al. [13] apply texture synthesis methods based on non-parametric sampling. Methods based on growing the known region into the missing region by continuing isophote lines discontinued by the border of the missing region are demostrated in [12].

Bornard et al. [15] draw motivation from texture synthesis based algorithms. They provide a spatio-temporal synthesis approach with coherence search to fill missing pixels in video frames.

Bertalmio et al. [14] discuss a simultaneous texture and structure image inpainting approach. Their approach is based on decomposing the image into a sum of two functions (also referred to as sub-frames). The first function represents the image structure and the other function represents its underlying texture. Image inpainting algorithm is applied on the structure information sub-frame and texture synthesis is applied on the sub-frame reflecting texture. Then the two reconstructed sub-frames are added to obtain the restored image.

Criminisi et al. [16] provide an exemplar based region filling and object removal method. Their idea is based on assigning priorities to determine the order of filling the missing pixels, propagating surrounding texture with structural information into the missing pixel region by setting confidence values to filled in pixels.

A low level global deterministic approach based on correspondence maps has been demonstrated by Demanet et al. [17]. A method based on texture generation inpainting using multiresolution analysis has been shown the work by Mustafa et al. [18].

5.3 Algorithm to resolve the problem of missing pixels

Our algorithm is divided into three stages as explained in the following sections:

5.3.1 Stage 1 - Mean and Median value interpolation

We make use of the capabilites of mean and median operators for image inpainting. These are simple operators exhaustively used in image processing and computer vision applications. Our method is based on growing the surrounding region of known pixels inwards towards the centre of the missing pixel region. The direction of growth is as shown in Fig 5.2. By filling in a set of missing pixels on the border of the unknown region at the first iteration, we fill the complete area of missing pixels through subsequent iterations. The procedure of filling missing pixels is based on two well known methods of image filtering - Mean filter and Median filter. We use mean and median values of a set of neighboring pixels in a 3x3 region to interpolate a value for the missing centre pixel. The methods are addressed as Mean Value Interpolation Method (MeanVIM) and Median Value Interpolation Method (MedianVIM) respectively. We assume that the exact positions of missing pixels is provided as an input to the algorithm.

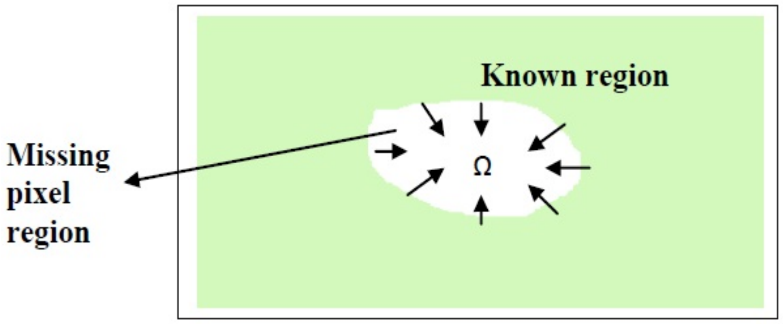


Figure 5.2: Diagram illustrating the direction of region growth for missing pixels. Ω denotes the missing pixel region.

We also assume image smoothness in all 3x3 regions over the image. We use the above methods to fill in a missing pixel having at least 4 known neighbors in its 3x3 neighbourhood. Experiments have shown that interpolation (MeanVIM or MedianVIM) based on less than 4 known neighbours does not produce satisfactory results.

To decide between MeanVIM and MedianVIM we perform pre-processing of the image. In the pre-processing step we evaluate a goodness measure of using MeanVIM and MedianVIM. The goodness measure is the mean square error between known pixels and their estimated value using MeanVIM or MedianVIM. The goodness measure is calculated for every value of number of known neighbors (mimimum 4) in a 3x3 neighborhood over the known region. In the known pixel region every pixel will have 8 known neighbors. Thus, to evaluate the goodness measure in case of four known neighbors we randomly choose a set of 4 neighbors for every known centre pixel. Next, we estimate its value using MeanVIM and MedianVIM and compute the error. The error is defined as the absolute difference between the estimated value and the actual value. Using this error for every known pixel we compute the Mean Squared Error (MSE) over the known pixel region. Thus, MSE measurements in each case of four to eight known neighbors is performed and the method that offers a lower MSE in each case is noted. This information is used during the initial stage of our algorithm.
For example Fig 5.3 shows the MSE evaluation of MeanVIM and MedianVIM over the tampered image of a crowd (see Fig 5.1). We observe that values of MSE for MeanVIM are higher compared to MedianVIM in cases where the number of known neighbor pixels are greater then 5. Thus, for filling missing pixels with more than 5 known neighbors we will use MedianVIM, otherwise MeanVIM is used.

Fig 5.4 shows the image of the crowd after performing MeanVIM/MedianVIM procedure over the tampered image shown in Fig 5.1.

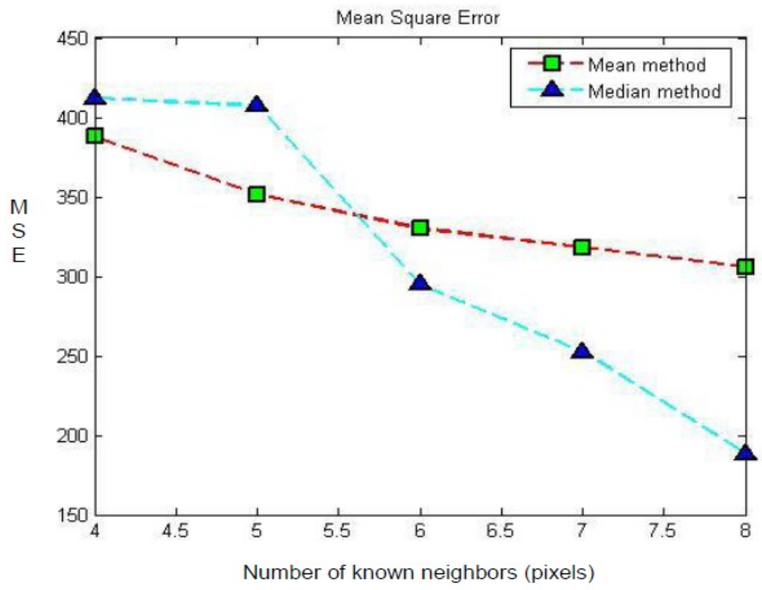


Figure 5.3: Graph illustrating the MSE with respect to MeanVIM and MedianVIM for Fig 5.1

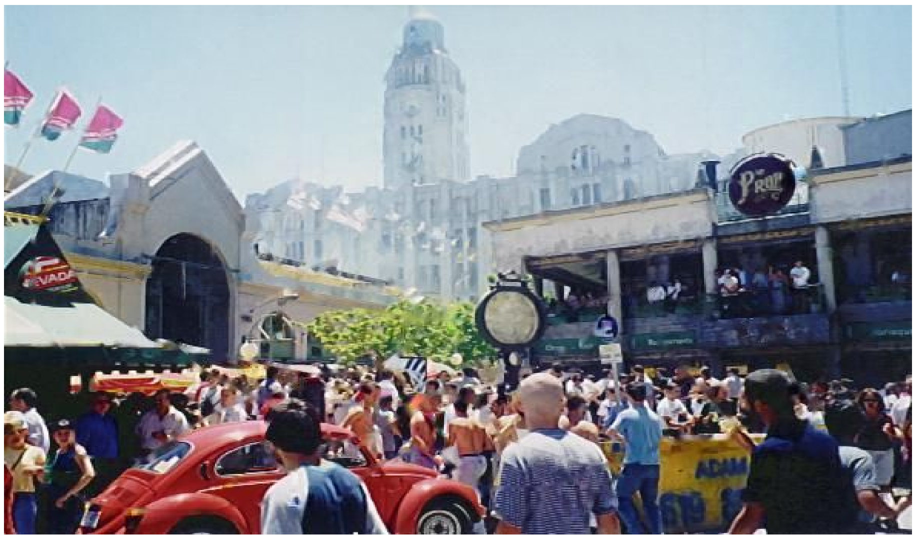


Figure 5.4: Resultant image after performing MeanVIM/MedianVIM procedure over tampered image in Fig 5.1

5.3.2 Stage 2 - Image smoothing

The image smoothing procedure involves repeated median filtering (size 3x3) over the filled in region (inpainted region). In this process the centre pixel is omitted in the 3x3 region. This results in noise filtering along with conserving the edges in the inpainted region. The number of times the median filtering has to be applied repeatedly has been experimentally determined to be around 20 after visual verification of the result on a set of test images.

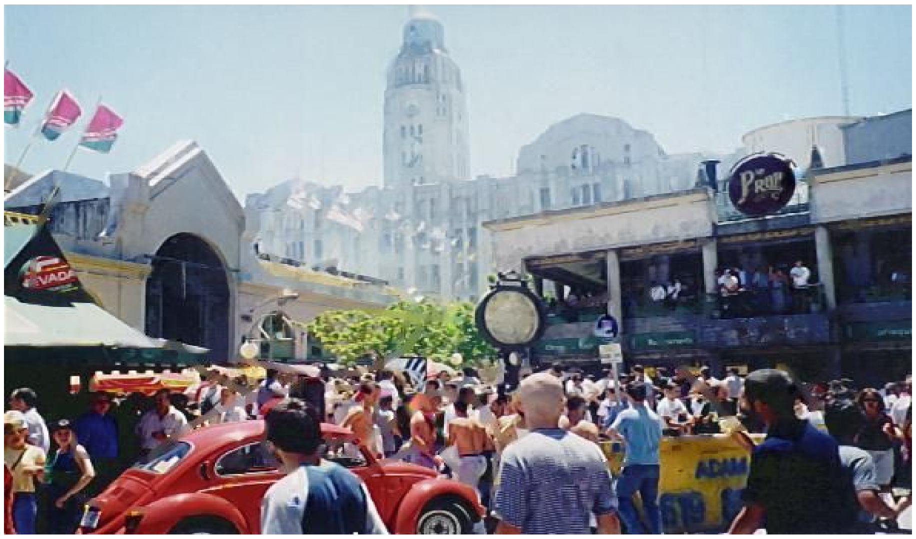


Figure 5.5: Resultant image after performing image smoothing operation on image in Fig 5.4

The result of image smoothing operation on crowd image after performing MeanVIM/MedianVIM procedure is shown in Fig 5.5

5.3.3 Stage 3 - Border blending

The last step of the algorithm is border blending procedure. This is done so as to ensure a smooth blending at the border of the known and inpainted regions. Border pixels in the context of our algorithm are defined as those pixels in the known region with atleast one missing pixel in its 3x3 neighborhood. For the purpose of border blending we make use of the median filter. Experiments have shown that it is best to run the border blending operation on only a few selected border pixels with a high gradient magnitude value. We use the Sobel filter to determine the gradient magnitude value at a pixel. To select the set of pixels for border blending, we first compute the histogram of gradient magnitude values of the border pixels. Next, we determine a threshold value of the gradient magnitude such that all border pixels above this value will be considered for border blending. Experiments have shown that it is best to choose top 40 percent of the border pixels with highest gradient magnitude measures. The experimental results can be best verified via visual analysis of the resulting images.

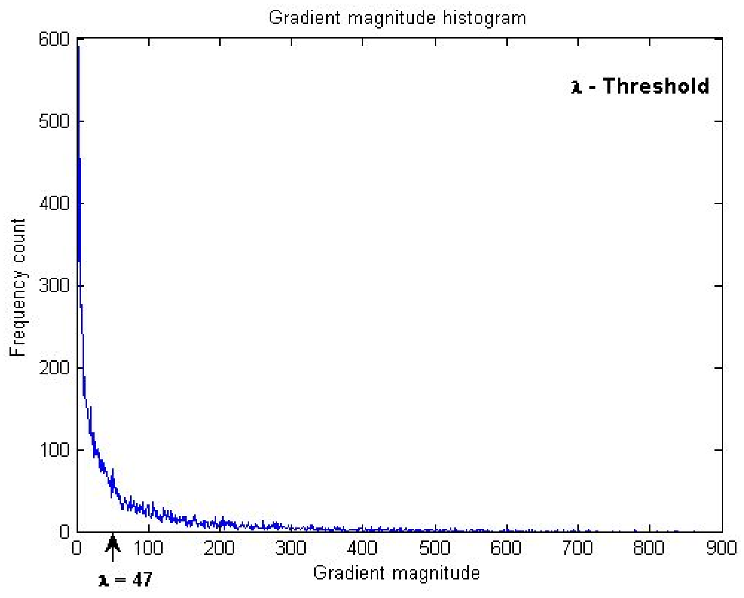


Figure 5.6: Histogram of the gradient magnitude of border pixels. All pixels with a gradient magnitude value ≥ Threshold, will be subjected to median filtering

Fig 5.6 shows the gradient magnitude histogram of the border pixels evaluated over the smoothed image in Fig 5.5. The threshold value determined is 47. All border pixels with a gradient magnitude value greater than or equal to the determined threshold value will undergo median filtering.

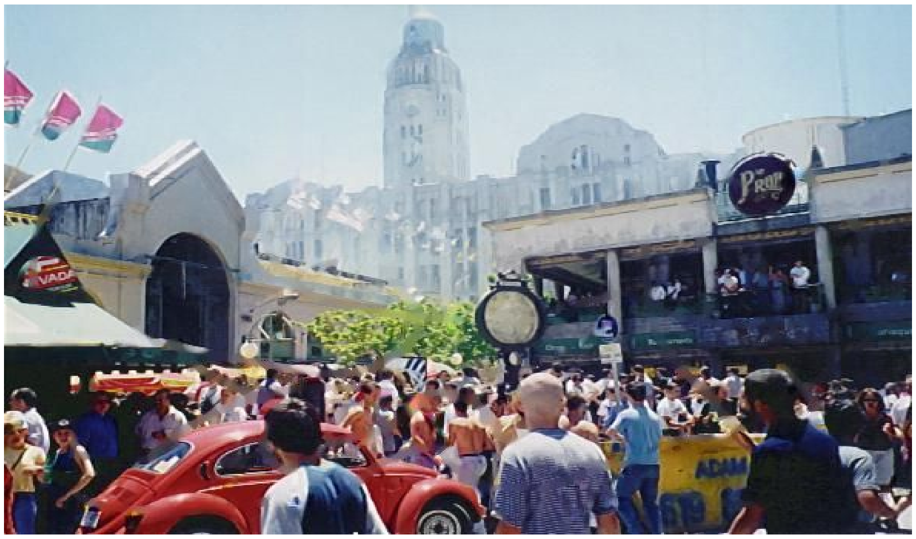


Figure 5.7: Resultant image after performing border blending operation over the smoothed image in Fig 5.5

Fig 5.7 shows the final restored image after performing border blending on smoothed image in Fig 5.5.

We can visually observe the improvement in the quality of the restored image through the results of different stages in our algorithm.

5.4 Inpainting software

We have built a software program that takes as input the image to be inpainted and the mask image that clearly identifies the missing pixels. The output of the program is the inpainted image (restored image) along with the MSE graph and gradient magnitude histogram graph. The mask image must be of the same size as the original image with missing pixels. The pixels corresponding in location to the missing pixels (in the original image) must be denoted by the intensity value 255 in the mask image. The remaining pixels in the mask image must be assigned a value 0.

5.5 Summary of the steps involved in implementing our algorithm

Steps involved in our algorithm to resolve the problem of missing pixels are as follows:

1. Pre-processing: The given image is scanned in the known regions to evaluate the goodness mesasure of the MeanVIM and MedianVIM methods in five different cases of the number of known neighbors ranging from 4 to 8.

2. Mean/Median value interpolation: Every pixel in the unknown region (location provided by the mask image) is checked for having 4 or more neighbors. All pixels that qualify this condition shall be filled by using the mean or median value of its known neighbors (based on goodnees measure). The remaining unknown pixels shall be filled in subsequent iterations. For every subsequent iteration the interpolated pixels in the previous iteration shall be included in the known region.

3. Image smoothing/denoising: A 3x3 median filter is iteratively run on the inpainted region excluding the centre pixel.

4. Border blending: We blend the border pixels which are found in the immediate surroundings of the unknown region. To do this we first evaluate the histogram

of the gradient magnitude value of border pixels. We then select a certain percentage (determined experimentally) of border pixels with highest gradient magnitude and run a median filter over them.

In our algorithm we make use of simple methods based on median and mean filters. We utilize the gradient magnitude histogram information for border blending. These methods are computationally simpler compared to any of the methods described in [12, 13, 14, 15, 16, 17, 18, 19, 20, 21]. We can also observe that the computation of all unknown pixels considered in one iteration of step 2 of the algorithm can be done in parallel. This is possible as no interpolated value in an iteration will be used in the same iteration by MeanVIM or MedianVIM.

5.6 Experiments and results

We present the results of our algorithm on a set of both natural and synthetically generated test images with missing pixels. Some of the test images have been taken from [3] with written permission from the author.

5.6.1 Test image obtained by scanning a photograph that has been damaged due to natural deterioration

The images in Fig 5.8 and Fig 5.9 show the damaged image and its respective mask image.

Fig 5.10 represents the MSE evaluation graph obtained from the damaged image. This gives us the information of which method namely MeanVIM or MedianVIM is better with respect to the number of known neighboring pixels. With this information

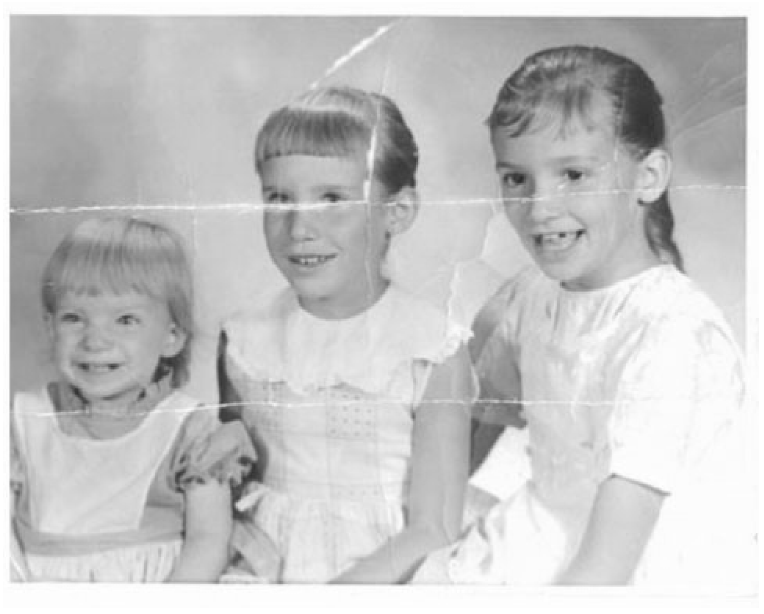


Figure 5.8: Damaged image showing 3 girls. Source: [3]

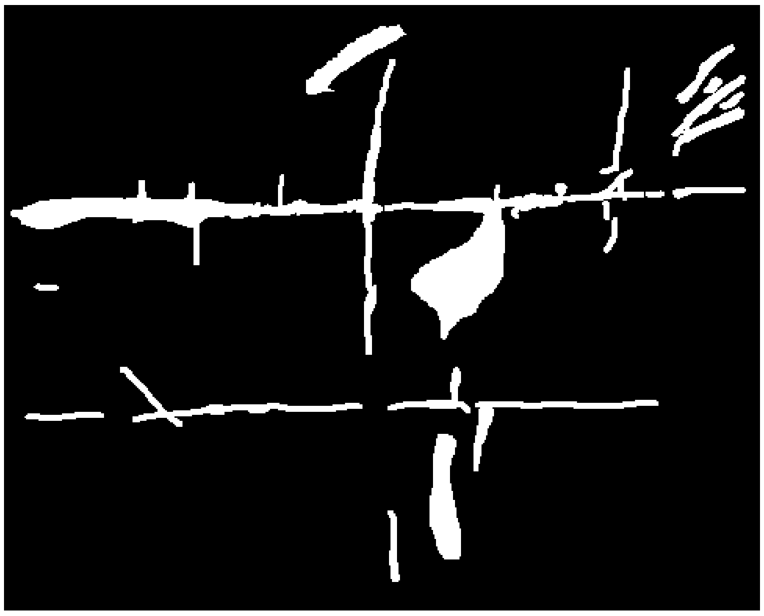


Figure 5.9: Mask image corresponding to damaged image shown in Fig 5.8.

we proceed with step 2 of our inpainting algorithm (see section 5.5). The results after completion of step 2 of our algorithm which fills in the missing pixel region is shown in Fig 5.11.

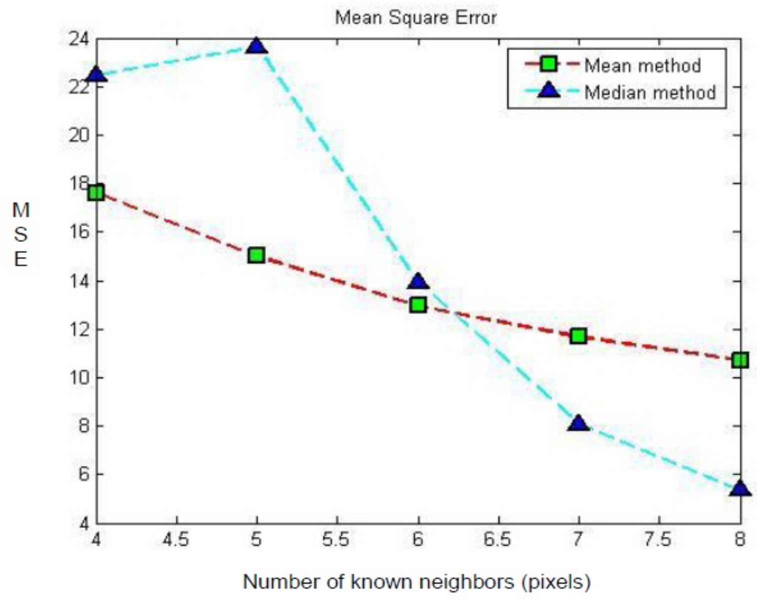


Figure 5.10: Graph representing the MSE evalution for image in Fig 5.8

Fig 5.12 is the result of image smoothing operation described in step 3 of our algorithm (see section 5.5).

Next we execute the final procedure of border blending on the image obtained after the smoothing step. The result after border blending is shown in Fig 5.14. The selection of threshold using gradient magnitude histogram is shown in Fig 5.13.

We find that the visual quality of the restored image by our algorithm is equally good compared to the result demostrated in [12].

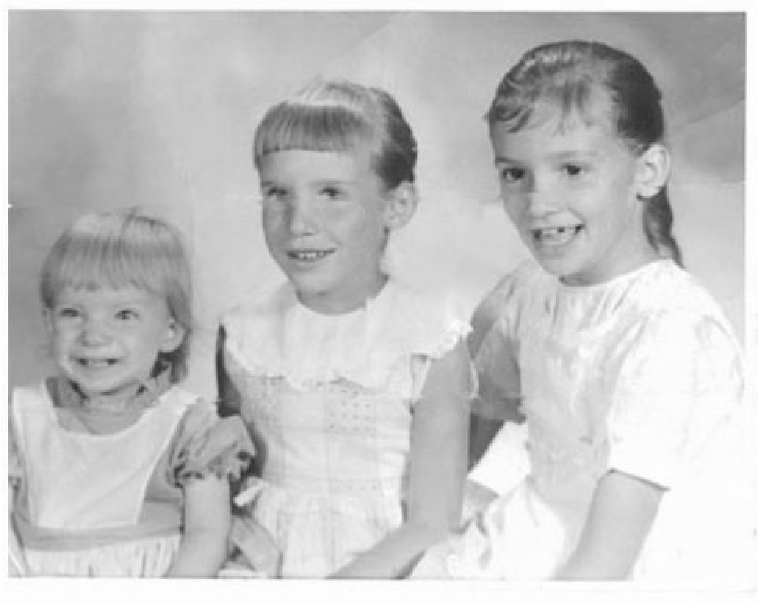


Figure 5.11: Image obtained after MeanVIM/MedianVIM (step 2 of our algorithm) on image in Fig 5.8

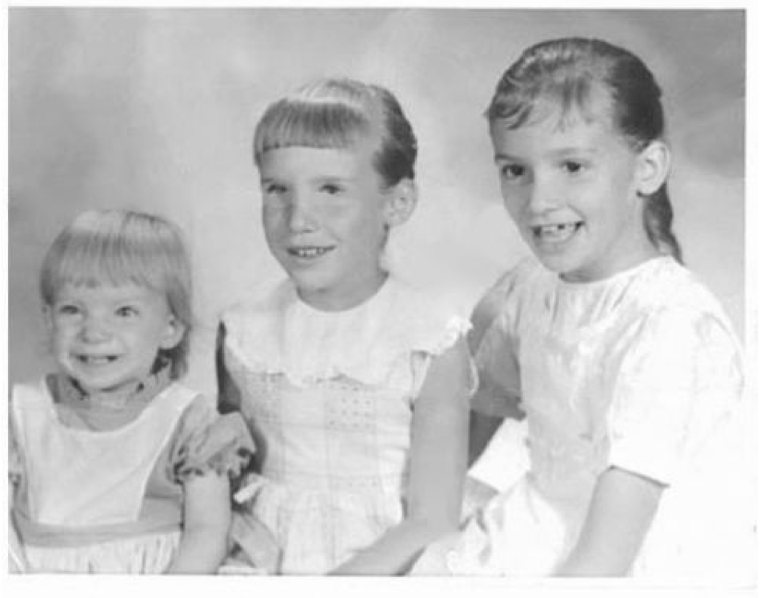


Figure 5.12: Image obtained after smoothing operation (step 3 of our algorithm) on image in Fig 5.11

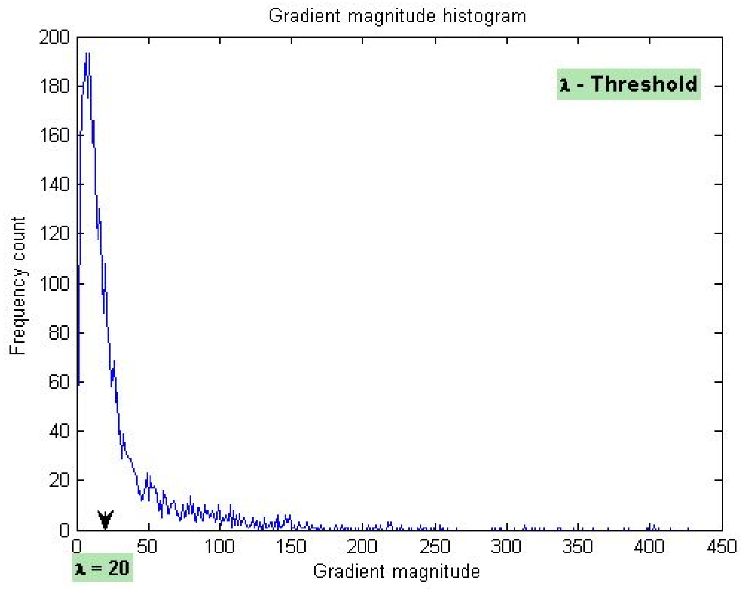


Figure 5.13: Gradient magnitude histogram and thresholding for border blending

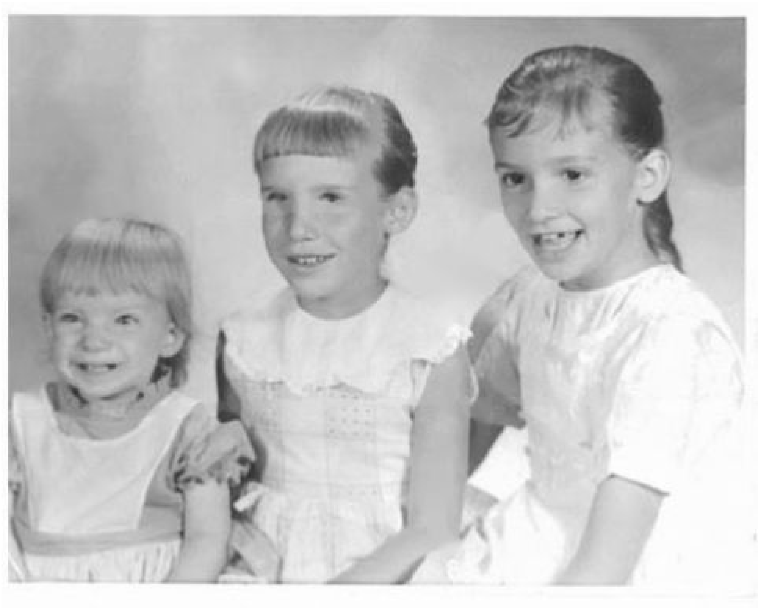


Figure 5.14: Image obtained after border blending procedure (step 4 of our algorithm) on image in Fig 5.12

5.6.2 Tampered test image synthetically generated by software

Fig 5.15 shows the image which has been tampered using a software program and the corresponding restored image using our algorithm.

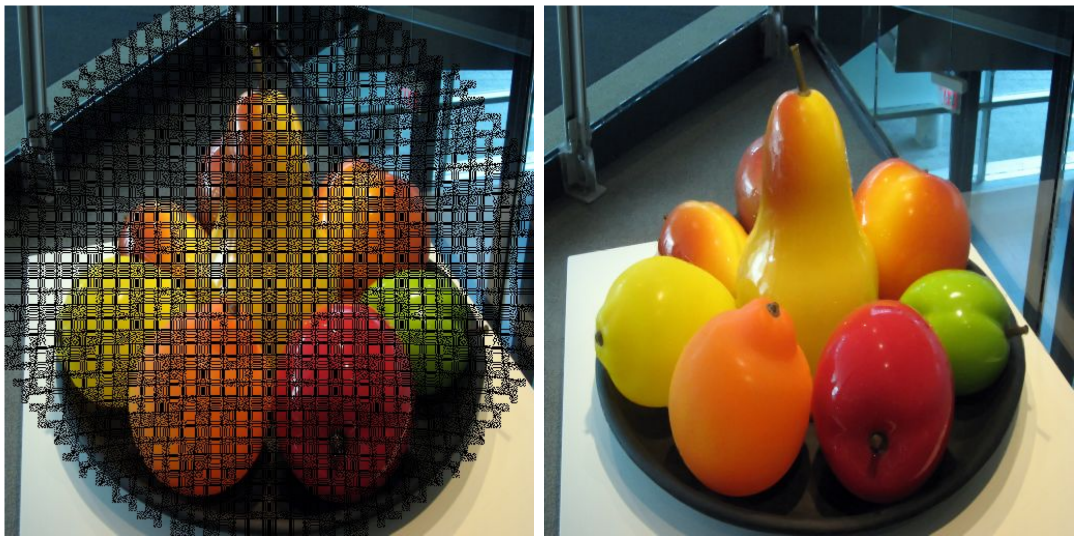


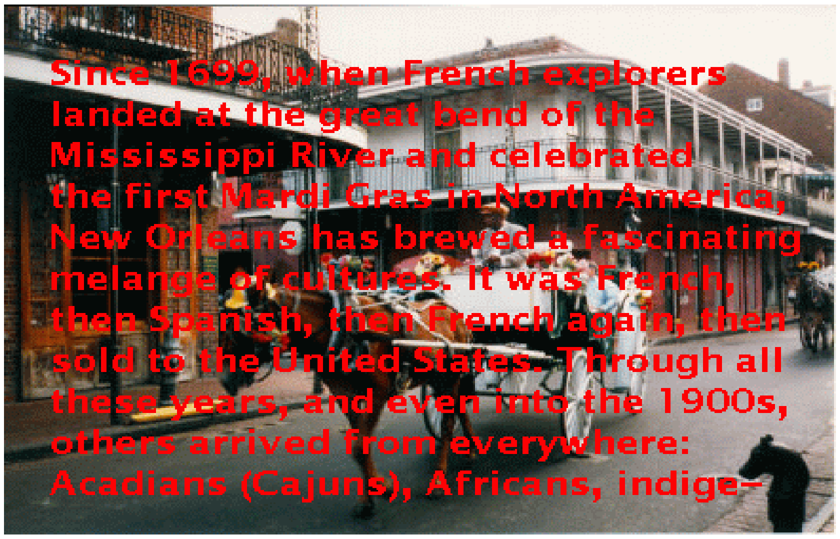
Figure 5.15: Left: Tampered image with missing pixels. Right: Restored image using our algorithm

5.6.3 Example of object removal in a tampered image

Fig 5.16 shows the tampered image of a horse cart where the region of missing pixels is considered as the areas covered by the english alphabets imprinted in red.

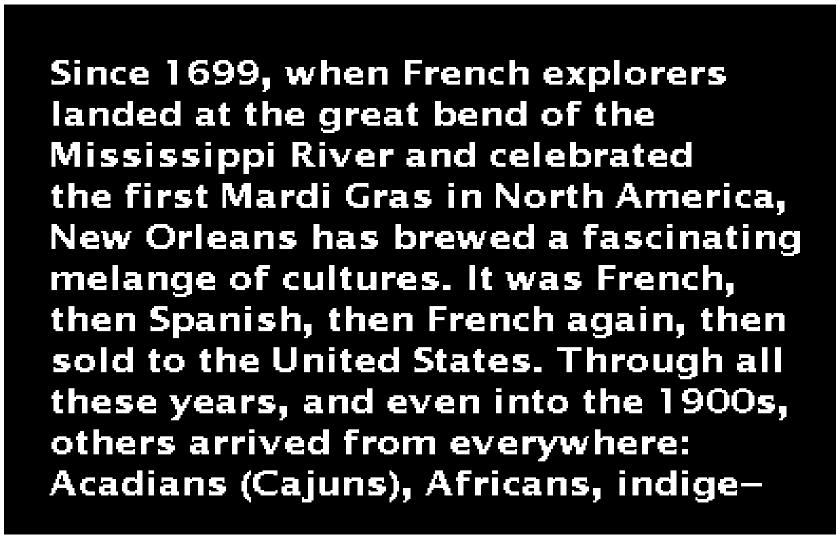
The associated mask image is shown in Fig 5.17.

The restored image is shown in Fig 5.18.



Since 1699, when French explorers
landed at the great bend of the
Mississippi River and celebrated
the first Mardi Gras in North America,
New Orleans has brewed a fascinating
melange of cultures. It was French,
then Spanish, then French again, then
sold to the United States. Through all
these years, and even into the 1900s,
others arrived from everywhere:
Acadians (Cajuns), Africans, indige–

Figure 5.16: Tampered image of horse cart. Alphabets imprinted in red represent areas of missing pixels



Since 1699, when French explorers
landed at the great bend of the
Mississippi River and celebrated
the first Mardi Gras in North America,
New Orleans has brewed a fascinating
melange of cultures. It was French,
then Spanish, then French again, then
sold to the United States. Through all
these years, and even into the 1900s,
others arrived from everywhere:
Acadians (Cajuns), Africans, indige–

Figure 5.17: Mask image corresponding to Fig 5.16

5.6.4 Example of object removal in a natural image

Fig 5.19 shows the image of a jumper. Here, the removal of the rope object tied to the jumper's ankle needs to be established. The removal of the rope object should

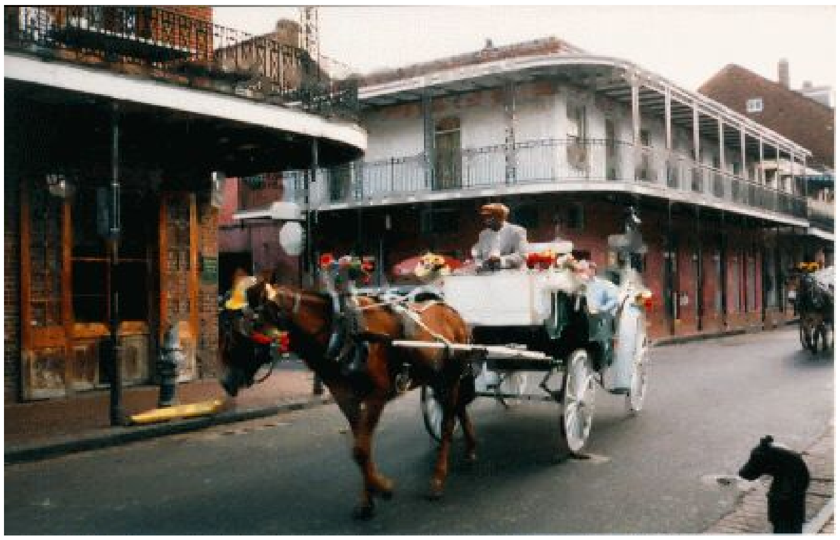


Figure 5.18: Restored image corresponding to the tampered horse cart image in Fig 5.16

be imperceptible in the resultant image. Fig 5.20 shows the resultant image using our algorithm.

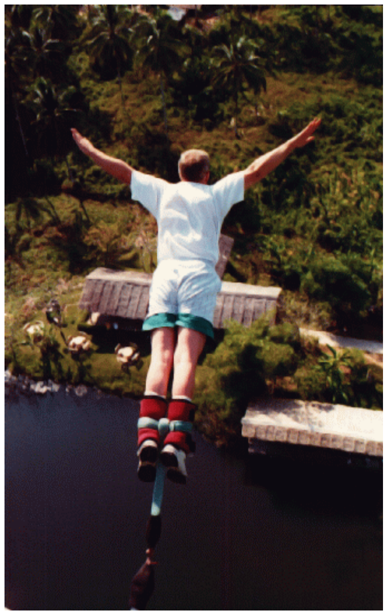


Figure 5.19: Original image of a jumper where the rope object needs to be removed

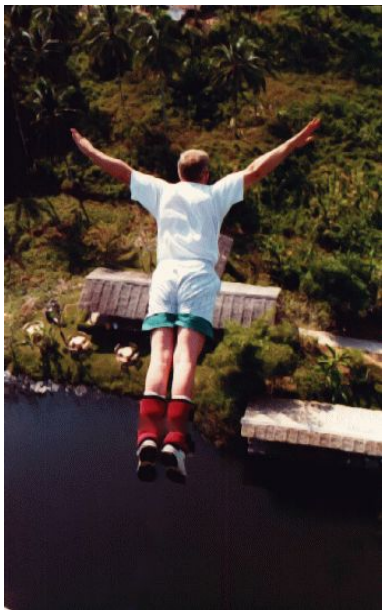
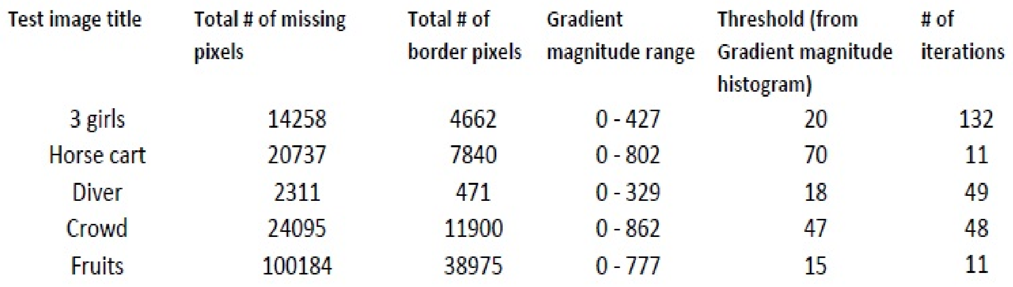


Figure 5.20: Processed image of the jumper where the rope object has been removed

5.6.5 Logistics from test images

Fig 5.21 shows the details of the number of missing pixels, number of border pixels, gradient magnitude threshold and the number of iterations of step 2 of our algorithm with respect to the test images that we have used.



Test image title	Total # of missing pixels	Total # of border pixels	Gradient magnitude range	Threshold (from Gradient magnitude histogram)	# of iterations
3 girls	14258	4662	0 - 427	20	132
Horse cart	20737	7840	0 - 802	70	11
Diver	2311	471	0 - 329	18	49
Crowd	24095	11900	0 - 862	47	48
Fruits	100184	38975	0 - 777	15	11

Figure 5.21: The table shows logistics based on our algorithm and test images

5.6.6 Limitations of the algorithm

The simplicity of our algorithm is also accompanied by a set of limitations. This is because of the fact that we do not exploit the structure and texture information in an image.
The MeanVIM and MedianVIM use only a 3x3 neighborhood. A larger neighborhood under consideration may result in better interpolation as we can use more known region data in the image. However, larger neighborhood consideration will result in higher computation especially in case of MedianVIM method.

5.7 Summary

In this chapter we have proposed a novel and simple scheme to resolve the problem of missing pixels under different scenarios. Our method is based on repeated and intelligent usage of the mean and median filters. Test results have been presented.

Chapter 6

Conclusions and Future scope

Through this thesis we have presented algorithms for generation of shift variant defocus and motion blurred images of moving 3D objects. We consider different geometrical aspects of the the 3D object being imaged and its motion parameters. 3D shapes used in our experiments are cylinder, sphere, cone, ellipsoid, etc. We precisely take into account the rigid body motion of these 3D objects in any arbitrary direction. Camera parameters such as aperture diameter, focal length and the location of image detector are used to calculate the blur circle radius of Point spread functions (PSFs) modeled by Gaussian and Cylindrical functions. We also describe a novel and simple method of image inpainting used for filling the missing pixels that arise due to round off errors that occur during interpolation or changes in magnification. Through our software SCO-3D, we have demonstrated the results that prove the correctness of our algorithms. The results can be used as test data with known ground truth in the testing and evaluation of image and video de-blurring algorithms. These methods can also be used to generate highly realistic 3D animation video.

We plan to expand our algorithms and software in the future to support additional features such as:

1. Combination of several shapes into one image and controlling their respective position via software.

2. Efficient calculation of forward blurring process using Rao transforms [22].

3. Using an accurate model of motion blur accounting for non-uniform motion of objects.

4. Generating blurred images using shift variant PSF from wave optics, in the presence of aberrations [4].

These additions will help the on-going research in the field of image restoration and deblurring. It will also provide versatile test data for research in the field of shape from defocus and shape from motion.

Bibliography

[1] Brian G. Schunck Ramesh Jain, Rangachar Kasturi, *Machine Vision*, McGraw-Hill, Inc., 1985.

[2] Jenn-Kwei Tyan, "Analysis and Application of Autofocusing and Three-Dimensional Shape Recovery Techniques based on Image Focus and Defocus," 1997, Phd Thesis, Department of Electrical and Computer Engineering, Stony Brook University.

[3] M Bertalmio, "Image Inpainting," *Website: http://www.tecn.upf.es/ mbertalmio/restoration0.html*, 2000.

[4] Shekhar Sastry, "Primary Abberations: An Investigation from the Image Restoration Perspective," 2009, Masters Thesis, Department of Electrical and Computer Engineering, Stony Brook University.

[5] M. Subbarao, "Determining Distance from Defocused Images of Simple Object," Tech. Rep. 89.07.20, Computer Vision Laboratory, Dept. of Electrical Engineering, State University of New York, 1989.

[6] M. Subbarao, "Efficient Depth Recovery through Inverse Optics," in *Machine Vision for Inspection and Measurement*, H. Freeman, Ed., pp. 101–126. Academic Press, New York, 1989.

[7] Gopal Surya, "Three Dimensional Scene Recovery from Image Defocus," 1997, Phd Thesis, Department of Electrical and Computer Engineering, Stony Brook University.

[8] Richard E. Woods Rafael C. Gonzalez, *Digital Image Processing*, Pearson Education, Inc., 2008.

[9] Michael Potmesil and Indranil Chakravarty, "Synthetic Image Generation with a Lens and Aperture Camera Model," *ACM Transactions on Graphics*, vol. 1, no. 2, pp. 85–108, 1982.

[10] Ming-Chin Lu Murali Subbarao, "Computer Modelling and Simulation of Camera Defocus," *Proceedings of SPIE*, vol. 1822, pp. 110–120, 1992.

[11] Michael Potmesil and Indranil Chakravarty, "Modeling Motion Blur in Computer-generated Images," *SIGGRAPH Comput. Graph.*, vol. 17, no. 3, pp. 389–399, 1983.

[12] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester, "Image Inpainting," in *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 2000, pp. 417–424, ACM Press/Addison-Wesley Publishing Co.

[13] Alexei A. Efros and Thomas K.Leung, “Texture Synthesis by Non-Parametric Sampling,” in *ICCV: International Conference on Computer Vision*, 1999, vol. 2, pp. 1033–1039.

[14] G. Sapiro M. Bertalmio, L. Vese and S. Osher, “Simultaneous Structure and Texture Image Inpainting,” *IEEE Trans. Image Process. 12*, pp. 882–889, 2003.

[15] Raphaël Bornard, Emmanuelle Lecan, Louis Laborelli, and Jean-Hugues Chenot, “Missing Data Correction in Still Images and Image Sequences,” in *MULTIMEDIA ’02: Proceedings of the tenth ACM international conference on Multimedia*, New York, NY, USA, 2002, pp. 355–361, ACM.

[16] K. Toyama A. Criminisi, P. Perez, “Region Filling and Object Removal by Exemplar-Based Image Inpainting,” *IEEE Transactions on Image Processing*, vol. 13, pp. 1200–1212, 2004.

[17] Laurent Demanet, Bing Song, and Tony Chan, “Image Inpainting by Correspondence Maps: a Deterministic Approach,” Tech. Rep., UCLA, 2003.

[18] D. Davcev B. Mustafa, V. Trajkovik, “Missing Data Correction in Still Images Using Multi-Resolution Analysis,” *Journal of Computing and Information Technology*, vol. 15, pp. 1–6, 2007.

[19] Pavlos Stavrou, Pavlos Mavridis, Georgios Papaioannou, Georgios Passalis, and Theoharis Theoharis, “3D Object Repair Using 2D Algorithms,” pp. 271–278, 2006.

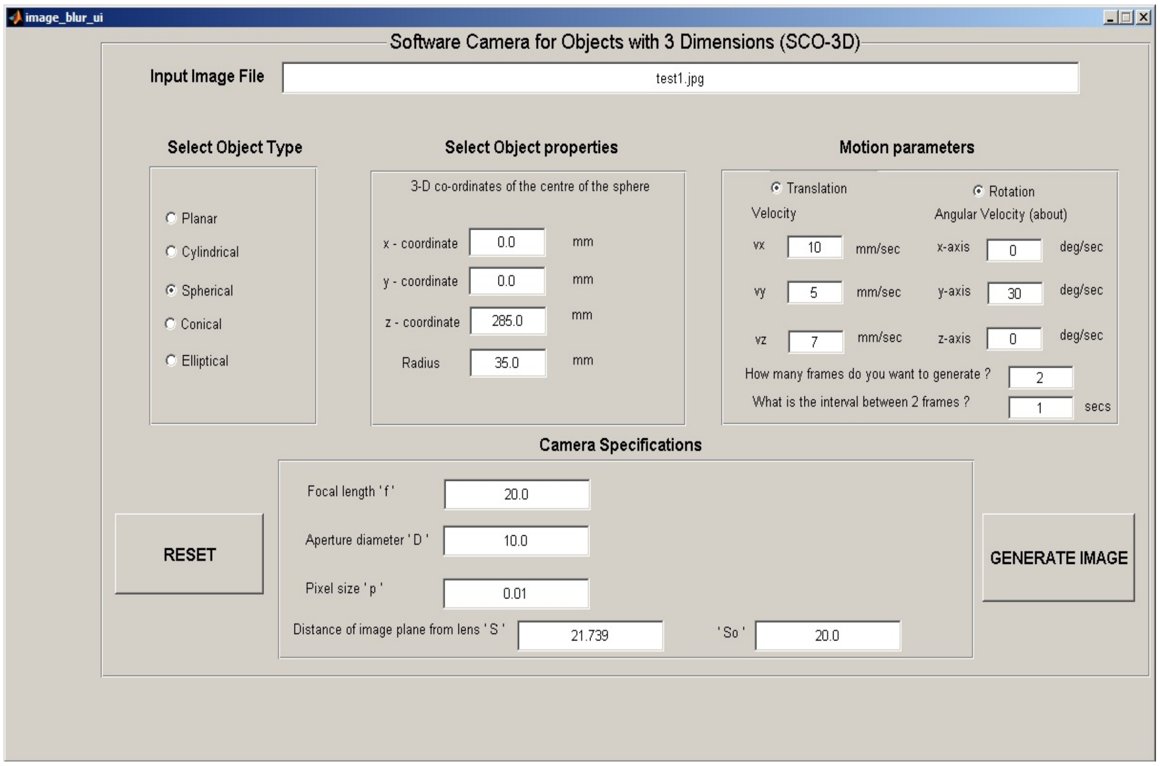
[20] A.C. Kokaram, R.D. Morris, W.J. Fitzgerald, and P.J.W. Rayner, "Interpolation of Missing Data in Image Sequences," *IEEE Transactions on Image Processing*, vol. 4, no. 11, pp. 1509–1519, 1995.

[21] Jianhong (Jackie) Shen, "Inpainting and the Fundamental Problem of Image Processing," *SIAM News*, vol. 36, no. 5, pp. 1–4, 2003.

[22] M. Subbarao, Y. Kang, S. Dutta, and X. Tu, "Localized and Computationally Efficient Approach to Shift-Variant Image Deblurring," in *IEEE International Conference on Image Processing*, 2008, pp. 657–660.

Appendix A

Fig 6.1 shows the screen shot of the user interface (UI) of our software SCO-3D.



image_blur_ui

Software Camera for Objects with 3 Dimensions (SCO-3D)

Input Image File: test1.jpg

Select Object Type

Planar
Cylindrical
[✓ Spherical]
Conical
Elliptical

Select Object properties

3-D co-ordinates of the centre of the sphere

x - coordinate	0.0	mm
y - coordinate	0.0	mm
z - coordinate	285.0	mm
Radius	35.0	mm

Motion parameters

[✓ Translation] [✓ Rotation]

Velocity			Angular Velocity (about)		
vx	10	mm/sec	x-axis	0	deg/sec
vy	5	mm/sec	y-axis	30	deg/sec
vz	7	mm/sec	z-axis	0	deg/sec

How many frames do you want to generate ? 2

What is the interval between 2 frames ? 1 secs

Camera Specifications

Focal length ' f ' 20.0

Aperture diameter ' D ' 10.0

Pixel size ' p ' 0.01

Distance of image plane from lens ' S ' 21.739 ' So ' 20.0

Figure 6.1: Screen shot of the user interface of SCO-3D

The UI mainly consists of five different panels that are used for setting different parameters for the program and two buttons to click.

Panels and their specifications are listed are follows:

1. Input Image File: The path or name of the input image file can be specified which is used as the focused image.

2. Select Object Type: One among the following list of object shapes can be selected. (a) Planar (b) Cylindrical (c) Spherical (d) Conical (e) Elliptical

3. Select Object Properties: The object properties panel changes its attributes according to the object selected (in Select Object Type panel). User is allowed to manually enter the values related to the 3D position of the object and its shape attributes.

4. Motion parameters: Translational and/or rotational motions can be chosen for the object. The velocities of the object in x, y, z direction and the angular velocity about x, y and z axes can be set in the given text boxes. We can choose the number of frames to be generated and the inter-frame interval.

5. Camera Specifications: Different camera parameters can be set as per the requirements.

6. Reset button: When clicked resets the panels listed in items 2 to 5 to their default values.

7. Generate button: When clicked generates and stores the resultant images (video frames) based on the settings.