ABSTRACT

A Direct Method for Surface Structure Recovering Based on UOFF

By
Ping Lin

The unified optical flow field (UOFF) theory which can be used for estimating motion and recovering surface structure was recently established in [9, 10]. The direct method developed in [2, 3, 4, 6, 7] does not need to explicitly solve the optical flow field and to find feature correspondence. Based on the UOFF, a direct method in space domain is developed to reconstruct the curved surface structure characterized by an Nth degree polynomial equation from a pair of stereo images. The initial work on this new method was reported in [8, 11].

In this study, I basically work on simulation images characterized by a 2nd degree polynomial equation. The main difference from the simulation results obtained in [8, 11], is that each object image in the pair of stereo images is formed from a set of images, which is referred to as *the composite image*. The gray levels in real images taken with general CCD cameras usually range from 0 to 255, since most of CCD camera systems are 8-bit in quantization resolution. This gray level range is too narrow to use the direct method in space domain to recover surface structure. However, with the Composite Image, it is possible to build a system with the current technology in the solid state industry, which is described in this thesis, to recover curved surface structures from real image sequences.

A DIRECT METHOD FOR SURFACE STRUCTURE RECOVERING BASED ON UOFF

by
Ping Lin

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science

Department of Electrical and Computer Engineering

January, 1993

APPROVAL PAGE

A Direct Method for Surface Structure Recovering
Based on UOFF

Ping Lin

Dr. Yun-Qing Shi, Thesis Adviser
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Edwin Hou, Committee Member
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. David Wang, Committee Member
Associate Professor of Computer and Information Science, NJIT

Blank Page

BIOGRAPHICAL SKETCH

Author: Ping Lin

Degree: Master of Science in Electrical and Computer Engineering

Date: January, 1993

Undergraduate and Graduate Education:

- Master of Science in Electrical and Computer Engineering
  New Jersey Institute of Technology, Newark, NJ, 1993

- Bachelor of Science in Electrical Engineering,
  The East of China Normal University, Shanghai, P. R. China, 1985

Major: Electrical and Computer Engineering

This thesis is dedicated to

my father Mr. Chenzhen Lin and my aunt Mrs. Fang Lin

ACKNOWLEDGMENT

The author would like to extend his gratitude to many people who have helped and contributed significantly in the bringing about this thesis. In particular, he would like to thank Dr. Yun-Qing Shi, his thesis advisor, for his ever present support and follow up, both technically, and interpersonally. He is also thankful to Dr. Edwin Hou, and Dr. David Wang for their valuable advice and assistance in completing this thesis.

The author would also like to express his thanks to many members of the Electronic Imaging Laboratory of Electrical and Computer Engineering Department, including: Mr. Jing-Ning Pan, Miss. Ying Lu, Mr. Bo-Lang Li, Mr. Feng You for their valuable help and support.

The author gratefully thanks his brother Mr. Hong Lin, who has given so much helps in the last two years of his graduate study.

TABLE OF CONTENTS

LIST OF TABLE

Table **Page**

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Estimation of motion and structure from image sequences has come to play an important role within the computer vision community over last ten years [12]. Basically there are two different approaches to recovering the structure of objects and the relative motion between objects and cameras: (1) the feature based approach [13] and (2) the optical flow based approach [5].

On one hand, it is known that extracting and establishing feature correspondence are difficult and only partial of solutions suitable for simplistic situation have been developed [1, p.299]. On the other hand, the optical flow approach needs to determine the optical flow field as an intermediate step. It involves a large amount of computation. Moreover, with only one equation and two unknowns, an extra constraint has to be imposed. The smoothness constraint of the optical flow field is one commonly utilized. However, sometimes it is not realistic.

The newly developed direct method [2, 3, 4, 6, 7], does not need to explicitly solve the optical flow field nor have to find feature correspondence. It is therefore very attractive.

However, this method only solves the planar surface successfully, because a set of linear equations can be formed from the minimization equation to recover motion and surface structure. But when surface order becomes higher, only non-linear equations can be formed from the minimization equation. It is really difficult to recover the motion and surface structure even for the second order surface.

Recently, a new concept of the unified optical flow field (UOFF) is established which is an extension of the fundamental optical flow formulation by

Horn and Schunck [5]. There are two major aspects of the UOFF concept: The first one is that the brightness function of an image is considered not only as function of time, but also as function of the various sensors' spatial positions. And the second one is that the brightness invariance equation is recognized not only for the time variation but also for the place variation. It is noted that the optical flow from a temporal image sequence discussed in [5] is a special case within the frame work of the UOFF.

In this thesis, the new concept of unified optical flow field is studied. Based on UOFF, a direct method in spatial case is applied in reconstructing a surface structure described by an Nth degree polynomial equation. A pair of stereo simulation images for a second order surface is used, and the Composite Image method is used to extend gray level range.

In Chapter Two, we first go through the development of the general case of brightness invariance equation. Then we discuss the UOFF, based on it, a direct method in space domain is developed. A set of linear equations are derived to determine all the coefficients of the polynomial equation which characterizes the curved surface structure.

In Chapter Three, a pair of simulation stereo images for a second order surface is used to test this direct method. The simulation images are created with two different methods. In the first method each image in the stereo image pair is created by a brightness function, whereas, in the second method each image is formed from a set of thus generated images with some weights and it is referred to as a Composite Image. By constructing the Composite Image, we can extend the image gray level range.

In Chapter Four, discussions are conducted and conclusions are drawn.

CHAPTER II

UOFF AND DIRECT METHOD

As mentioned earlier, the UOFF approach is a relatively new method that combines parameter's time, and space (the sensor spatial position) in the brightness function. In this chapter, the united optical flow field (UOFF) will be introduced, and based on UOFF the direct method in space domain for recovering surface structure will be developed.

2.1 United Optical Flow Field (UOFF)

2.11 Image Space

Consider a sensor located in a specific position in 3-D world space keeps generating images about the scene. As time goes by, the sensor forms a sequence of images at this particular position in 3-D space. The set of these images can be represented with brightness function g(x, y, t), where x and y is coordinates on the image plane. This is the basic outline about brightness function g(x, y, t) which is treated by Horn and Schunck [1].

A different sequence of images can be formed as follows. At a specific moment in time, there are infinitely many sensors in the Imaging space to view the object from all possible different positions, then we cannot use the previous brightness function g(x, y, t) to describe the gray levels of the image plane. Combining the two factors of time and space, we obtain yet another, much larger, set of images. To describe the brightness of this new set, we could use a more general brightness function:

$$g = g(x, y, t, s) \tag{2.1}$$

where s indicates the sensor's position in 3-D world space, i.e., the coordinates of the sensor center and the orientation of the optical axis of the sensor. Since the sensor as a solid object can be translated (which has three degrees of freedom) and rotated (which has two degrees of freedom), s is a 5-D vector. That is

$$s = (x, y, z, \beta, \gamma) \tag{2.2}$$

where x, y and z represent the coordinate of the optical center of the sensor in 3-D world space; and β, γ represent the orientation of the optical axis of the sensor in 3-D world space.

2.1.2 Brightness Invariance Equation (BIE)

In the image space, each image points is corresponding to an arbitrary but fixed point P. At time t, in 3-D world space possess the same brightness, i.e., P is isotropic.

For a point P in world coordinate system, if its optical radiation is invariant with respect to a time interval from t_1 to t_2, we then have:

$$g(x_p(t_1,s_1), y_p(t_1,s_1), t_1, s_1) = g(x_p,(t_2,s_1), y_p(t_2,s_1), t_2, s_1) \tag{2.3}$$

This is the brightness time-invariance equation and it is utilized in the determination of optical flow by Horn and Schnook [1]. At a specific moment t_1, if the optical radiation of P is isotropic, we then get:

$$g(x_p(t_1,s_1), y_p(t_1,s_1), t_1, s_1) = g(x_p,(t_1,s_2), y_p(t_1,s_2), t_1, s_2) \tag{2.4}$$

This is the brightness space-invariance equation. If the two variables, time and space, are considered simultaneously, we get the brightness time-and-space invariant equation, i.e.,

$$g(x_p(t_1,s_1), y_p(t_1,s_1), t_1, s_1) = g(x_p,(t_2,s_2), y_p(t_2,s_2), t_2, s_2) \tag{2.5}$$

Comparing two brightness functions g(x(t, s), y(t, s), t, s) and g(x(t+Δt, s+Δs), y(t+Δt, s+Δs), t +Δt, s+Δs), Δt, the variation of time, and Δs, the variation in spatial position of sensor, are very small. Due to the time-and-space-invariance of brightness, we can get:

$$g(x(t, s), y(t, s), t, s = g(x(t+\Delta t, s+\Delta s), y(t+\Delta t, s+\Delta s), t+\Delta t, s+\Delta s) \quad (2.6)$$

The right-hand side of the above equation is expanded into the Taylor series. It leads to

$$g(x(t+\Delta t, s+\Delta s), y(t+\Delta t, s+\Delta s), t+\Delta t, s+\Delta s) = g(x(t, s), y(t, s), t, s) +$$

$$\frac{\partial g}{\partial x}\left(\frac{\partial x}{\partial t}\mathrm{dt} + \frac{\partial x}{\partial s}\mathrm{ds}\right) + \frac{\partial g}{\partial y}\left(\frac{\partial y}{\partial t}\mathrm{dt} + \frac{\partial y}{\partial s}\mathrm{ds}\right) + \frac{\partial g}{\partial t}\mathrm{dt} + \frac{\partial g}{\partial s}\mathrm{ds} + \varepsilon \qquad (2.7)$$

where ε contains the second and higher order terms in t and/or Δs. The next equation follows then from the use of Equation (2.3)

$$\left(\frac{\partial g}{\partial x}\mathrm{u} + \frac{\partial g}{\partial y}\mathrm{v} + \frac{\partial g}{\partial t}\right)\Delta t + \left(\frac{\partial g}{\partial x}\mathrm{u}^{s} + \frac{\partial g}{\partial y}\mathrm{v}^{s} + \frac{\partial g}{\partial s}\right)\Delta s + \varepsilon = 0 \qquad (2.8)$$

where $u = \frac{\partial x}{\partial t}$, $v = \frac{\partial y}{\partial t}$, $u^{S} = \frac{\partial x}{\partial s}$, $v^{S} = \frac{\partial y}{\partial s}$. Dividing both sides of the above equation by Δt, ignoring the term containing and examining the limit as Δt→0 yields,

$$\frac{\partial g}{\partial t} + \frac{\partial g}{\partial x}\left(\frac{\partial x}{\partial t} + \frac{\partial x}{\partial s}\frac{\delta s}{\delta t}\right) + \frac{\partial g}{\partial y}\left(\frac{\partial y}{\partial t} + \frac{\partial y}{\partial t}\frac{\delta s}{\delta t}\right) + \frac{\partial g}{\partial s}\frac{\delta s}{\delta t} = 0 \qquad (2.9)$$

where $\frac{\delta s}{\delta t} = \lim_{\Delta t \to 0}\frac{\Delta s}{\Delta t}$

Denote the velocity of a point in the image space by

$$V\left(\frac{\mathrm{dx}}{\mathrm{dt}}, \frac{\mathrm{dy}}{\mathrm{dt}}, \frac{\delta s}{\delta t}\right)$$

where $\frac{d}{dt} = \frac{\partial}{\partial t} + \frac{\partial}{\partial s}\frac{\delta s}{\delta t}$ is a differential operator.

Let $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial s})$ be a vector operator in imaging space. Equation (2.9) then becomes

$$\frac{\partial g}{\partial t} + V \bullet \nabla g = 0 \tag{2.10}$$

Similar to the well-known continuity equation in fluid dynamics :

$$\frac{\partial g}{\partial t} + \nabla \bullet (gV) = 0 \tag{2.11}$$

As compared with Equation (2.10), the left-hand side of Equation (2.9) lacks a term of $g\nabla \bullet V$. It is related to the sum of all the second and higher order terms of t and/or s, i.e., the ε in the right-hand side of Equation (2.6).

2.1.3 United Optical Flow Field

As mentioned in the image space, the general brightness function is described in Equation 2.1

$$g = g(x, y, t, s) \tag{2.1}$$

And

$$s = (x, y, z, \beta, \gamma) \tag{2.2}$$

x, y and z represent the coordinate of the optical center of the sensor in 3-D world space; and β, γ represent the orientation of the optical axis of the sensor in 3-D world space.

In dealing with a "spatial" sequence, we consider the various positions of the cameras in space at a specific moment. The movements of imaging camera are

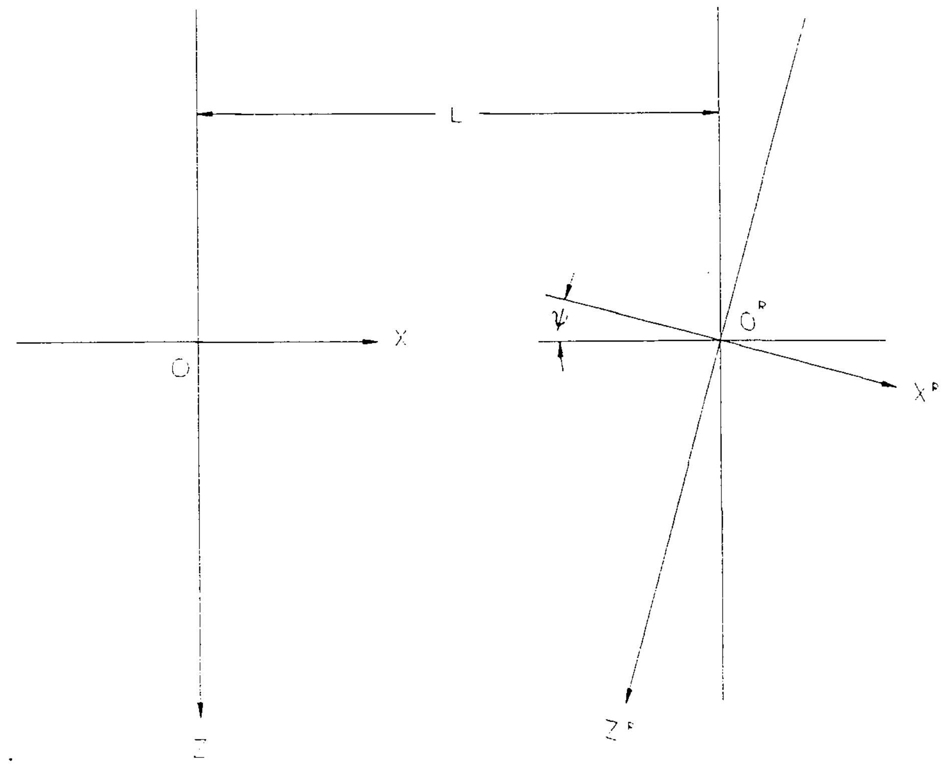

Figure 2.1 An image setting

described in Figure 2.1. The right camera is moving where the left camera is fixing in space. The movement of the right camera can be viewed as: the translation of the lens center O^R followed by a rotation of the optical axis O^RZ^R. The two optical axes OZ and O^RZ^R are assumed, for simplicity, to be coplanar. The lens center O^R can therefore only be translated on the OXZ plane. Hence any translation of the O^R on the OXZ plane can be decomposed as the translation along the direction parallel to the OX axis and the translation along the direction parallel to the OZ axis. The rotation of optical axis O^RZ^R about the O^RY^R is marked by ψ. However, the assumption made previously that the O^R lies on the OX implies $z = 0$. Therefore z will not be considered under the assumption made.

Define

$$\delta s = (x^2 + \chi^2 \psi^2)^{1/2}$$

χ is a characteristic length chosen according to imaging setting. So δs is a measurement of a variation of the right camera position with respect to the left camera position. i.e. the variation of the position of the right lens center O^R with respect to that of the left optical axis OZ. Let s denote the camera position in space and its superscript denotes which camera is considered. For instance, s^L is used to denote the left camera position, s^R the right camera position, and we have $s^R = s^L + \delta s$. It is obvious that when $x = 0$, $\psi = 0$ (hence $\delta s = 0$), the two cameras are at the same position in space. i.e., $s^L = s^R$. If the camera's moving path is specified on the x-ψ plan, different values of x and ψ (hence different values of δs) determine the various values of s^R. i.e., the various positions of the right camera in space.

At special moment t_1, if the optical radiation of a point P is isotopic we then get Equation (2.4)

$$g(x_p(t_1,s_1), y_p(t_1,s_1), t_{1,}s_1) = g(x_p,(t_1,s_2), y_p(t_1,s_2), t_1,s_2) \qquad (2.4)$$

the images generated with sensors at different spatial positions can be viewed as a space sequence of images. Since at this situation, the vairation of time is equal to zero, the brightness time-and-space invariant Equation (2.8) will reduce to:

$$\frac{\partial g^L}{\partial x}u^s + \frac{\partial g^L}{\partial y}v^s + \frac{\partial g^L}{\partial s} = 0 \qquad (2.12)$$

Now let us take a close look at each quantity in Equation (2.12). The quantities with the superscript L are related to the left sensor. The $\frac{\partial g^L}{\partial s}$ can be estimated from the image data as following:

$$\frac{\partial g}{\partial s} = \frac{g^R(x^L, y^L, t) - g^L(x^L, y^L, t)}{\delta s} \tag{2.13}$$

The quantities with the superscript R are related to the right sensor. The u^S and v^S are defined as follows.

Let

$$dx = x^R - x^L$$
$$dy = y^R - y^L$$

where (x^R, y^R) and (x^L, y^L) are projections of the same world point on the right and the left image planes, respectively. δx and δy are therefore, respectively, the horizontal and vertical coordinate differences of the image points, corresponding to the same world point in 3-D space, reflected on the right and left image planes. And then,

$$u^s = \lim_{\delta s \to 0} \frac{\delta x}{\delta s} \tag{2.14}$$

$$v^s = \lim_{\delta s \to 0} \frac{\delta y}{\delta s} \tag{2. 15}$$

Hence, u^s and v^s defined above are the spatial variation rates of δx and δy with respect to δs. These two quantities generated from the spatial sequence of images, can be viewed as counterparts of u^L and v^L, (or u^R and v^R), generated from a temporal sequence of images.

2.2 Direct Method in Space Domain

2.2.1 Relationships between 3-D Space and Image Plane

Following our system setup, the left sensor is located at the origin of Cartesian coordinate system, and the other sensor is at a known different position. Since any rigid body motion can be resolved into two components, a translation and a rotation, the right sensor movement with respect to the origin of the world coordinate center, can be also decomposed into translational component T_S and rotational component ω_S. The subscript s indicates the s-domain. As described in Figure 2.2.

$$T_s = (U_s, V_s, W_s)^T \tag{2.16}$$

where the superscript T represents the transposition of the concerned vectors, U_s, V_s, W_s are translation velocity component along with the OX, OY, OZ, direction, respectively.

$$\varpi_s = (A_s, B_s, C_s)^T \tag{2.17}$$

A_s, B_s, C_s are rotation velocity component around the OX, OY, OZ, directions, respectively.

Since the object movement is equivalent to the camera movement in the reverse direction, the object movement with respect to the origin of the world coordinate center, V_s, can be also decomposed into translational component $-T_S$ and rotational component $-\omega_S$ due to the movement of right camera. Let r_s be a vector $(X, Y, Z)^T$. By rigid body moving equation, we have

$$V_s = -T_s - \varpi \times r_s \tag{2.18}$$

if we define

$$V_s = (\frac{dX}{ds}, \frac{dY}{ds}, \frac{dZ}{ds})^T \tag{2.19}$$

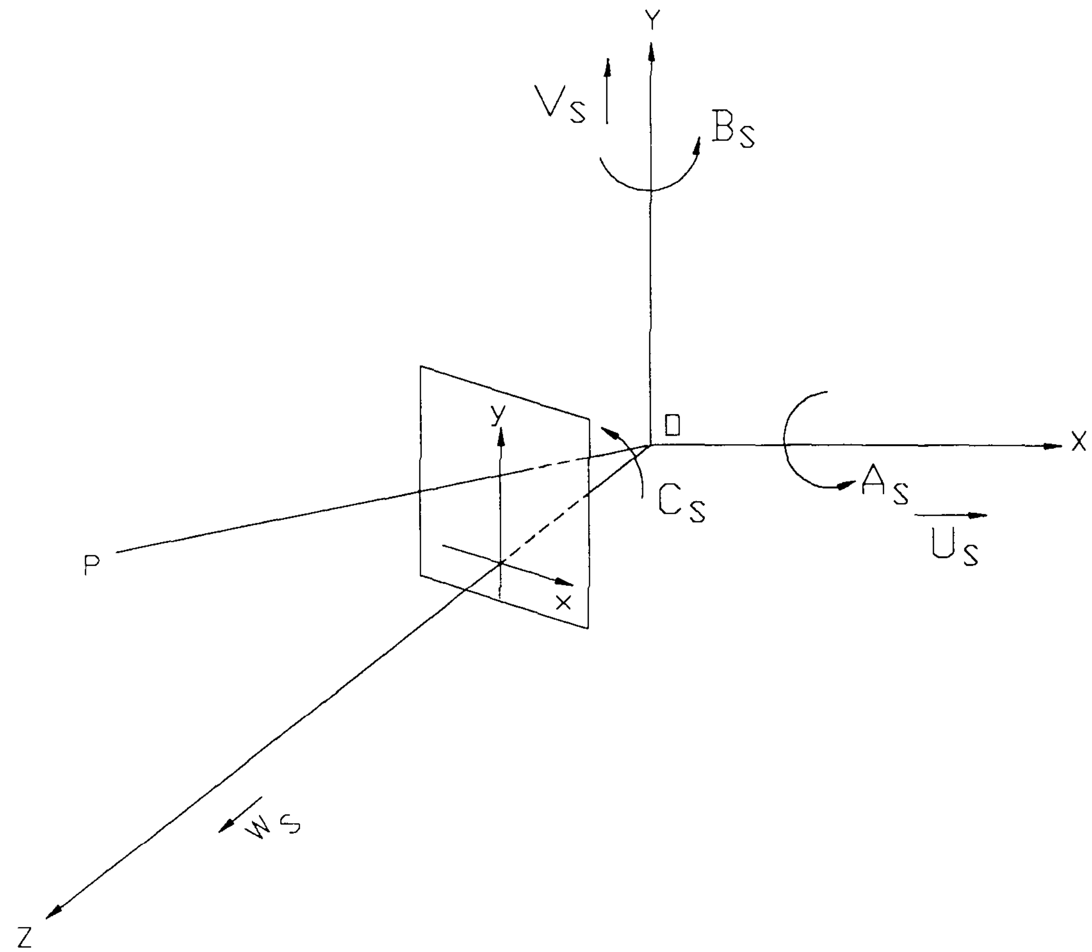

Figure 2.2 Cartesian coordinate system

Then Equation (2.18) can be rewritten in component form:

$$\frac{dX}{ds} = -U_s - B_s Z + C_s Y \tag{2.20}$$

$$\frac{dY}{ds} = -V_s - C_s X + A_s Z \tag{2.21}$$

$$\frac{dZ}{ds} = -Ws - AsY + B_s X \tag{2.22}$$

Reproducing the prospective projection formulas we have

$$x = \frac{X}{Z}$$

$$y = \frac{Y}{Z}$$

so

$$\frac{dX}{ds} = \frac{d}{ds}(xZ) = x\frac{dZ}{ds} + Z\frac{dx}{ds}$$

$$Z\frac{dx}{ds} = \frac{dX}{ds} - x\frac{dZ}{ds}$$

Combining this with Equation (2.20), we have:

$$u^s = \frac{dx}{ds} = \frac{\frac{dX}{ds}}{Z} - x\frac{\frac{dZ}{ds}}{Z} \tag{2.23}$$

and similarly with Equation (2.21):

$$\frac{dY}{ds} = \frac{d}{ds}(yZ) = y\frac{dZ}{ds} + Z\frac{dy}{ds}$$

$$\Rightarrow v^s = \frac{dy}{ds} = \frac{\frac{dy}{ds}}{Z} - \frac{Y\frac{dZ}{ds}}{Z^2} \tag{2.24}$$

From Equations (2.23) and (2.20) we get:

$$u_s = (-\frac{U_s}{Z} - B_s + C_s y) - x(-\frac{W_s}{Z} - A_s y + B_s x) \tag{2.25}$$

and from Equations (2.23) and (2.20) we get:

$$v_s = (-\frac{V_s}{Z} - A_s + C_s x) - y(-\frac{W_s}{Z} - A_s y + B_s x) \tag{2.26}$$

2.2.2 Surface Structures of Nth degree Polynomial Equation

The object under study in the 3-D world is a surface structure that can be described by an Nth degree Polynomial Equation in the O-XYZ coordinate system. The general form of the polynomial is

$$\sum_{j=0}^{K-1} \lambda_j X^{\alpha_j} Y^{\beta_j} Z^{\gamma_j} = 0 \tag{2.27}$$

where $0 \le \alpha_j + \beta_j + \gamma_j \le N$, and K is the number of coefficients present.

Out of the K different coefficients of the polynomial, there are K-1 independent terms, the polynomial can therefore be normalized with respect to one arbitrary coefficient $\lambda(r)$. By rewriting the polynomial as

$$\sum_{j=0, j\neq r}^{K-1} (\lambda_j X^{\alpha_j} Y^{\beta_j} Z^{\gamma_j}) + \lambda_r X^{\alpha_r} Y^{\beta_r} Z^{\gamma_r} = 0$$

we divide through both sides by term λ_r

$$\sum_{j=0, j\neq r}^{K-1} (\lambda_{nj} X^{\alpha_j} Y^{\beta_j} Z^{\gamma_j}) + X^{\alpha_r} Y^{\beta_r} Z^{\gamma_r} = 0$$

where $\lambda_{nj} = \dfrac{\lambda_j}{\lambda_r}$, λ_j is normalized.

This equation will be used in reconstructing the polynomial since now we have (K- 1) independent coefficients.

2.2.3 Direct Method in Space Domain

Let us examine Equations (2.25), (2.26) again. it is found that u^s, v^s are expressed as a linear function of the motion parameters A_s, B_s, C_s, A_s, B_s, C_s and only one factor Z^{-1}, since in spatial domain there is a brightness invariance equation.

$$\frac{\partial g}{\partial x} u^s + \frac{\partial g}{\partial y} v^s + \frac{\partial g}{\partial s} = 0$$

and if we define

$$g_x = \frac{\partial g^L}{\partial x}$$

$$g_y = \frac{\partial g^L}{\partial y}$$

$$g_s = \frac{\partial g^L}{\partial s}$$

so from Equation (2.25) and Equation (2.26), we get

$$\frac{1}{Z} = \frac{(-A_s y + B_s x)(x g_x + y g_y) - g_x(-B_s + C_s y) - g_s(-C_s x + A_s) - g_s}{x W_s g_x + y W_s g_y - U_s g_x - V_s g_y} \tag{2.28}$$

or if we define Q = l/Z,

$$Q = \frac{(-A_s y + B_s x)(x g_x + y g_y) - g_x(-B_s + C_s y) - g_s(-C_s x + A_s) - g_s}{x W_s g_x + y W_s g_y - U_s g_x - V_s g_y} \tag{2.29}$$

Note that A_S B_S C_S U_S V_S W_S can be determined once the relative positions of the two sensors in stereo imagery are known. Also, g_x, g_y, g_s can be determined from the given image data by similar algorithms as used in [1]. x, y are the coordinated on the image plane.

Substituting Q^{-1} for Z in Equation (2.27), we get

$$\sum_{j=0, j \neq r}^{K-1} \lambda_{nj} X^{\alpha_j} Y^{\beta_j} Q^{-\gamma_j} + X^{\alpha_r} Y^{\beta_r} Q^{-\gamma_r} = 0 \tag{2.31}$$

Now, to reconstruct the original polynomial we have to use the recovered depth as in Equation (2.31) so we define a performance function as

$$J = \iint_R \{ \sum_{j=0, j \neq r}^{K-1} \lambda_{nj} X^{\alpha_j} Y^{\beta_j} Q^{-\gamma_j} + X^{\alpha_r} Y^{\beta_r} Q^{-\gamma_r} \}^2 \, dxdy \tag{2.32}$$

where R is the region on the image plane associated with the concerned surface in 3-D space.

The task here is to find a set of coefficients λ_{nj}; so that the performance function is minimized (brought as close to zero as possible). It is well known that following linear equations are necessary conditions for minimization of the J function:

$$\frac{\partial J}{\partial \lambda_i} = 0 \tag{2.33}$$

where i = 0,1, 2, . . . , (K-1), and i $\neq$ r. Differentiating with respect to λ_i yields

$$\iint_R 2\{ \sum_{j=0, j\neq r}^{K-1} \lambda_{nj} X^{\alpha_j} Y^{\beta_j} Q^{-\gamma_j} + X^{\alpha_r} Y^{\beta_r} Q^{-\gamma_r} \} X^{\alpha_i} Y^{\beta_i} Q^{\lambda_i} dxdy = 0 \tag{2.34}$$

or,

$$\sum_{j=0, j\neq r}^{K-1} \iint_R \{\lambda_{nj} X^{\alpha_i + \alpha_j} Y^{\beta_i + \beta_j} Q^{-\gamma_i + \gamma_j}\} dxdy \lambda_j =$$

$$\iint_R \{\lambda_{nj} X^{\alpha_r + \alpha_i} Y^{\beta_r + \beta_i} Q^{-\gamma_r + \gamma_i}\} dxdy \tag{2 35}$$

with i = 0, 1, 2, . . . , (K-1), and i $\neq$r.

The above equations can be put in matrix form as follows:

$$M_{i,j} = \iint_R \{X^{\alpha_i + \alpha_j} Y^{\beta_i + \beta_j} Q^{-\gamma_i + \gamma_j}\} dxdy \tag{2 36}$$

$$D_i = \iint_R \{\lambda_{nj} X^{\alpha_r + \alpha_i} Y^{\beta_r + \beta_i} Q^{-\gamma_r + \gamma_i}\} dxdy \tag{2 37}$$

In this set of linear equations, all of the coefficients of the Nth degree polynomial,i.e., j, where i = 0,1, 2, . . ., (K-1), and i $\neq$ r are unknown. All of the entries in the matrix $M_{i,j}$ and in the vector D_i can be calculated from the given image data.

The structure of the surface can therefore be recovered, because the polynomial equation describing the surface has been fully determined.

CHAPTER III

SIMULATION AND RESULTS

To test the previous algorithm for recovering surface structure, a *C* program was written. The program can be divided into two parts:

1. Build a pair of stereo images of a surface and save them in image files of variable sizes (typically 128 by 128).
2. Deal with the image data created earlier, and attempt to recover the depth field at each image pixel by applying Equation (2.26), and then set up the K and D matrices, solve the (K-l) independent 's linear equations.

3.1 The Construction of Simulation Images

The simulation images can be created with two kinds of methods. For the first one, we assume a surface structure expressed by a certain polynomial equation and perform a prospective projection from the surface onto the two image planes, the left and the right. The second method, the first step repeats the first method and creates a couple pair of images and sums up each pixel value and then gets a pair of images.

3.1.1 Image setting

The geometry of the setup is a typical stereo system. There is a world Cartesian coordinate (X, Y, Z), and two identical cameras positioned as shown in Figure 3.1. In 3-D space, the movement of a camera can be divided into three translational components and two degrees rotational components. The rotation of a camera around its optical axis is not considered since no change in the image information will result. The optical axis of the left camera coincides with the Z axis of the world coordinate system, and the center of the left image plane is located at (0,0,1) exactly. The surface structure has its own coordinate system.

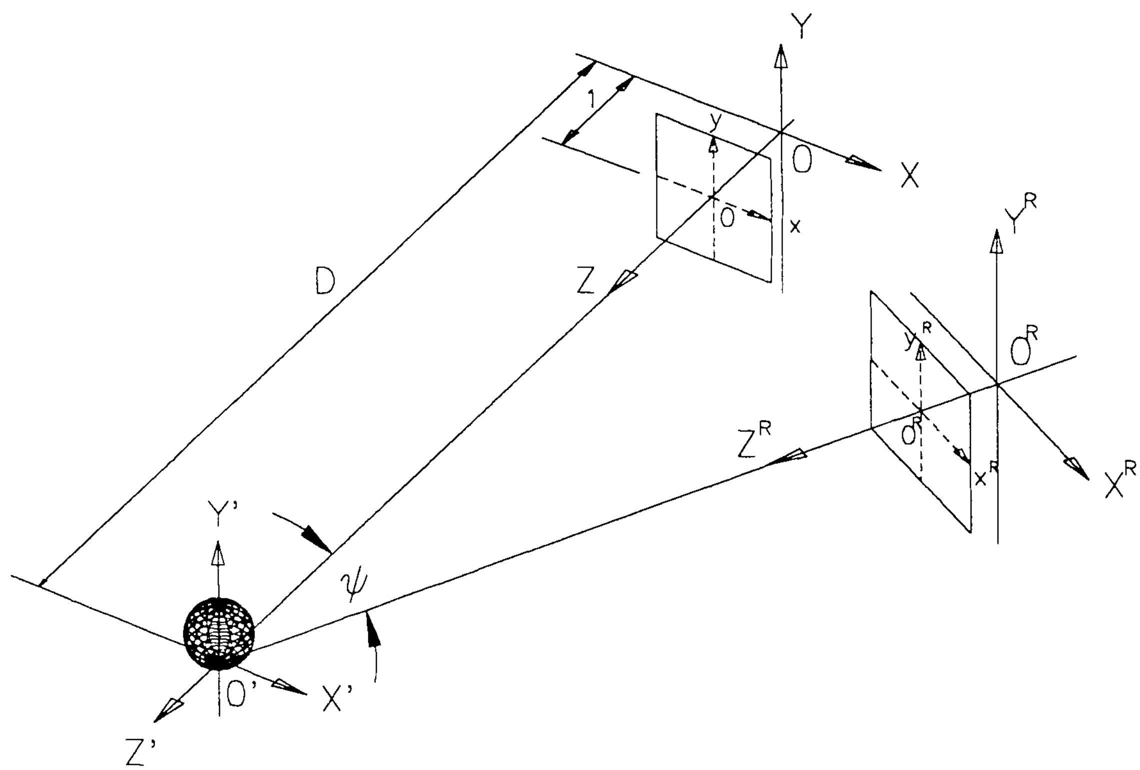

Figure 3.1 System setup

The origin of this system is O', which differs from O by a sole translation of distance D, in the positive direction of Z. This coordinate system is used to make the description of the surface structure easier later and to avoid numerical problems. The value of D is much larger than 1. For the right camera, which the image plane is rotated with ψ degrees, and translated in the X, and Z directions, maintains a constant distance, D, between the center of the right image plane and the origin of the surface structure.

Like this setup, we have

$$
\begin{aligned}
A_s &= 0.0 \\
B_s &= -\frac{\psi}{\delta s} \\
C_s &= 0.0 \\
U_s &= -D\frac{\sin\psi}{\delta s} \\
V_s &= 0.0 \\
W_s &= -D\frac{(1-\cos\psi)}{\delta s}
\end{aligned}
\tag{3.1}
$$

As Bs Cs are the components of the rotation rate vector of the right camera in the X, Y, and, Z directions, respectively . Us, Vs and, Ws are the components of the translation rate vector of the right camera in the X, Y,and, Z directions, respectively.

3.1.2 The Polynomial Equations

The Sphere chosen is described by the following polynomial:

$$X^2 + Y^2 + (Z-D)^2 - 16 = 0 \tag{3.2}$$

or in the shifted coordinates (Figure 3.2):

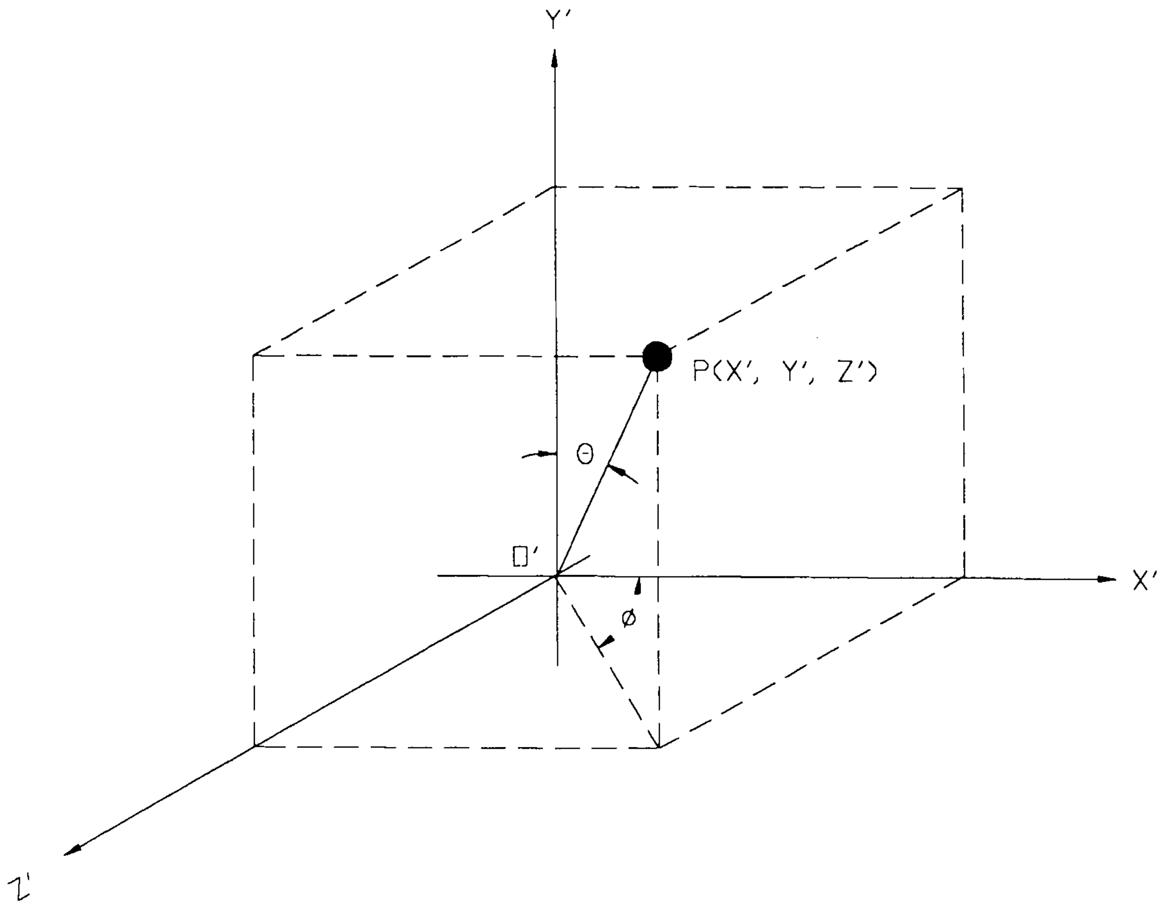

Figure 3.2 The surface in X', Y', Z' coordinates

$$X'^2 + Y'^2 + Z'^2 - 16 = 0 \tag{3.3}$$

This surface equation fit the general definition of the polynomial given by

$$\sum_{j=0}^{K-1} \lambda_j X^{\alpha_j} Y^{\beta_j} Z^{\gamma_j} = 0$$

To associate combinations of X, Y, Z, the standard α, β, γ are shown in Table 3.1. It is the one of assumption throughout our work. Any arbitrary combinations fit this table. When the polynomial equation becomes high degree, the higher degree terms can be added without changing the lower ones.

Table 3.1: The standard α, β, γ used

λ	α	β	γ
λ_0	0	0	0
λ_1	1	0	0
λ_2	0	1	0
λ_3	0	0	1
λ_4	2	0	0
λ_5	1	1	0
λ_6	1	0	1
λ_7	0	2	0
λ_8	0	1	1
λ_9	0	0	2
λ_{10}	3	0	0
λ_{11}	2	1	0
λ_{12}	2	0	1
λ_{13}	1	2	0
λ_{14}	1	1	1
λ_{15}	1	0	2
λ_{16}	0	3	0
λ_{17}	0	2	1
λ_{18}	0	1	2
λ_{19}	0	0	3

From Table 3.1, we know that for the Sphere:

$$\lambda_0 = -16.0, \quad \lambda_4 = 1.0, \quad \lambda_7 = 1.0, \quad \lambda_9 = 1.0.$$

3.1.3 The Construction of Simulation Images

Both surfaces are built in a similar manner. We start from the image screen, project a beam of light toward the object, and try to solve for the closest point in the object surface in which the beam actually hits the object surface. If no solution exists at all, then that particular pixel does not "see" the object surface, and is assigned a background gray level value. If solutions do exist, we choose the closest one to the image plane, and then from the knowledge of the coordinates in the space of that point, we assign a gray level value according to a generating function.

$$g = K_1 \cos(K_2\theta)\sin(K_3\phi) + K_1 \tag{3.4}$$

Where K_1, K_2, K_3 are some constants. The θ, ϕ are in spherical coordinates and defined in the Figure 3.2. The Cartesian coordinates transfer into spherical coordinates as follows:

$$\theta = \tan^{-1}\frac{(X'^2 + Y'^2)^{1/2}}{Y}$$

$$\phi = \tan^{-1}\frac{Z'}{X'}$$

The Left Image: The left sensor is aligned with the world coordinate system, and thus it is relative to a simple matter of projecting world points onto the screen. The distance D between the origin of the world coordinate and the origin of the shifted coordinate system is chosen to be 100.0 cm. The same length units are assumed for the surface, so we have a relative idea about the size of the surfaces being considered.

To maximize efficiency, the light from the object should occupy about 75% of the screen; so the screen size is taken to be 0.11 cm x 0.11 cm. The screen consists of I_n x J_n pixels. The row and column indices of the screen are represented by i and j. Coordinates of the image, x and y, are related linearly to i and j as follows:

$$x = \frac{0.11j}{J_n} + x_0$$

$$y = \frac{0.11i}{I_n} + y_0 \qquad (3.5)$$

where for the sphere, $x_0 = -0.055$ is the offset in the x direction, and, $y_0 = 0.050$ the offset in the y direction, from the equation we know the x moves in the same direction of j, while y moves in the opposite direction of i.

In building the left image, we have to start from the screen, not from the world point, and our algorithm goes as follows: For a point p(i, j) on the screen, the x, and y are calculated as in Equation (3.35). The surface equation for the Sphere surface in the world coordinates is

$$X^2 + Y^2 + (Z - D)^2 - 16 = 0 \qquad (3.6)$$

Where D is the distance between the origin of the world coordinate system and the origin shifted coordinate system. By prospective projection rule, X and Y are equalt to xZ and yZ, respectively. X, Y Substituted with xZ, yZ, in Equation (3.6), it becomes another second order equation:

$$x^2Z^2 + y^2Z^2 + (Z - D)^2 - 16 = 0 \qquad (3.7)$$

In this equation only Z is an unknown value, it can be very easily solved numerically. When Z is obtained, we get Z' = (Z-D), Y' = yZ, and X' = xZ. From X', Y', Z',we get θ and ϕ as shown above, and we obtain the gray level value from the function. The quantization value of g is then assigned to the pixel at (i, j) which we started with. This process is repeated for all the pixels of the screen. If it is not reasonable for the solution of Z at a pixel, the gray value of this pixel is assigned as background value.

The Right Image: The right sensor's position in 3-D space differs from that of the left sensor by a mere rotation of angle ψ about the O'Y' axis. The angle is positive when viewed from the positive Y' axis down onto the origin and rotates clockwise. The arm of rotation is D, and thus the distance between the camera and the world origin is kept the same. Furthermore, the relation between i and j, and x^R and y^R remains the same, namely,

$$x^R = \frac{0.11j}{J_n} + x_0 \tag{3.9}$$

$$y^R = \frac{0.11i}{I_n} + y_0$$

Note that the surface is symmetric about the O'Y' axis, the axis of rotation. Therefore, since both sensors stay coplanar, no change in the shape will take place in the right camera. The difference will come from the gray function values. The gray function will have to be rotated in the opposite direction for our simulation to be correct. So the same procedure is followed here to get X', Y', Z', but in the gray function ϕ is replaced with $(\phi+\psi)$. Following exactly the same procedure from here on, we build the right sensor image and save it in an image file.

3.2 Simulation

We now have a pair of stereo images for the analysis. By Equations (2.35), (2.36), (2.37), (2.29) the surface structure can be recovering directly from the image data.

$$\sum_{j=0, j\neq r}^{K-1} \iint_R \{\lambda_{nj} X^{\alpha_j+\alpha_i} Y^{\beta_j+\beta_i} Q^{-\gamma_j+\gamma_i}\} dxdy \lambda_j =$$

$$\iint_R \{\lambda_{nj} X^{\alpha_r+\alpha_i} Y^{\beta_r+\beta_i} Q^{-\gamma_r+\gamma_i}\} dxdy \tag{2 35}$$

$$M_{i,j} = \iint_R \{X^{\alpha_j+\alpha_i} Y^{\beta_j+\beta_i} Q^{-\gamma_j+\gamma_i}\} dxdy \qquad (2\ 36)$$

$$D_i = \iint_R \{\lambda_{nj} X^{\alpha_r+\alpha_i} Y^{\beta_r+\beta_i} Q^{-\gamma_r+\gamma_i}\} dxdy \qquad (2\ 37)$$

$$Q = \frac{(-A_s y + B_s x)(xg_x + yg_y) - g_x(-B_s + C_s y) - g_s(-C_s x + A_s) - g_s}{xW_s g_x + yW_s g_y - U_s g_x - V_s g_y} \qquad (2.29)$$

We are going to examine the terms that make up Equation (2.29), and describe how we can obtain all of them.

x and y: The x, y is the coordinate of each pixel in the image plane. Since Q is an array of the same size image screen I_n x J_n, the depth Z is computed for all the pixels of the screen. If we denote i to represent the row index, and j to represent the row index, and j to represent the column index, then we have, as a linear relation between the indices and the real values. The x and y can be calculated by Equation (3.5):

$$x = \frac{0.11j}{J_n} + x_0$$

$$y = \frac{0.11i}{I_n} + y_0$$

g_x and g_y: The g_x is the gradients in x direction, and g_y is in the y direction, We are interested in computing for a spatial sequence of images, not a temporal one. Horn and Scunck in [1] described a simple approximation of g_x and g_y. We slightly modified the approximations from.

It becomes:

$$\Delta g_j = 0.25\{L(i, j+1) - L(i, j) + L(i+1, j+1) - L(i+1, j) + R(i, j+1) - R(i, j) + R(i+1, j+1) - R(i+1, j)\} \tag{3.10}$$

$$\Delta g_i = 0.25\{L(i+1, j) - L(i, j) + L(i+1, j+1) - L(i, j+1) + R(i+1, j) - R(i, j) + R(i+1, j+1) - R(i, j+1)\} \tag{3.11}$$

where L(i, j) and R(i, j), are the gray values of the pixels at row i and column j, of the left, and right images, respectively. Considering the change in the values of x and y, from pixel to pixel. We have:

$$\Delta x_j = \frac{0.11}{I_n}$$

$$\Delta y_j = \frac{0.11}{J_n} \tag{3.12}$$

where I_n and J_n are the number of pixels in the row and the column, respectively. In this way, we can approximate $\frac{\partial g^L}{\partial x}$ and $\frac{\partial g^L}{\partial y}$ by:

$$\frac{\partial g}{\partial x} = \frac{\partial g}{\partial j}\frac{\partial j}{\partial x} = \frac{\Delta g_j}{\Delta x_j} \tag{3.13}$$

$$\frac{\partial g}{\partial y} = \frac{\partial g}{\partial i}\frac{\partial i}{\partial y} = \frac{\Delta g_j}{\Delta y_i} \tag{3.14}$$

g_s: g_s is a rate of change of spatial gradient. In order to get the rate of change of spatial gradient, we first get the change of spatial gradient, Δg_s. The approximate equation for Δg_s is:

$$\Delta g_s = 0.25\{L(i, j) - R(i, j) + L(i+1, j) - R(i+1, j) + L(i, j+1) - R(i, j+1) + L(i+1, j+1) - R(i+1, j+1)\} \tag{3.15}$$

Then we need to get the measure of the transition between the left and right cameras; the spatial transition becomes:

$$ds = (x'^2 + z'^2 + D^2\beta^2)^{1/2}$$

where x' and z', represent the displacement of the right optical center from that of the left optical center. Dβ is the length of the arc made by the rotation of the camera. In short, s is a measure of the movement from the first camera to the second, the "Spatial" movement. g_S is therefore approximated as:

$$g_s = \frac{\Delta g_s}{\delta s} \tag{3.16}$$

By Equation (2.36), and Equation (2.37)

$$M_{i,j} = \iint_R \{X^{\alpha_j+\alpha_i} Y^{\beta_j+\beta_i} Q^{-\gamma_j+\gamma_i}\} dxdy \tag{2 36}$$

$$D_i = \iint_R \{\lambda_{nj} X^{\alpha_r+\alpha_i} Y^{\beta_r+\beta_i} Q^{-\gamma_r+\gamma_i}\} dxdy \tag{2 37}$$

We can solve the $M_{i,j}$ and D_i, but the result is not good enough. Since the Q^{-1} is too big compared with the values of X and Y, the significance of the values of X and Y was lost. To prevent this situation, we watch Equation (2.30), and express the surface in the X', Y', Z' system, we have

$$\sum_{j=0,j\neq r}^{K-1} (\lambda_{nj} X'^{\alpha_j} Y'^{\beta_j} Z'^{\gamma_j}) + X'^{\alpha_r} Y'^{\beta_r} Z'^{\gamma_r} = 0$$

Going back to the X, Y, Z system, and replacing $(X', Y', Z')^T$ by $(X, Y, (Z\text{-}D))^T$, and then putting that into the above equation and substituting Q^{-1} for Z, we get.

$$\sum_{j=0,j\neq r}^{K-1} (\lambda_{nj} X^{\alpha_j} Y^{\beta_j} (Q^{-1} - D)^{\gamma_j}) + X^{\alpha_r} Y^{\beta_r} (Q^{-1} - D)^{\gamma_r} = 0 \tag{3.17}$$

where X = xZ, and Y = yZ.

Then the performance function is defined as:

$$J = \iint_R \{ \sum_{j=0, j \neq r}^{K-1} \lambda_{nj} X^{\alpha_j} Y^{\beta_j} (Q - D)^{-\gamma_j} + X^{\alpha_r} Y^{\beta_r} (Q - D)^{-\gamma_r} \}^2 dxdy \tag{3.18}$$

So we get the new M matrix

$$M_{i,j} = \iint_R \{ X^{\alpha_j + \alpha_i} Y^{\beta_j + \beta_i} (Q^{-1} - D)^{\gamma_j + \gamma_i} \} dxdy \tag{3.19}$$

and D matrix

$$D_i = \iint_R \{ \lambda_{nj} X^{\alpha_r + \alpha_i} Y^{\beta_r + \beta_i} (Q^{-1} - D)^{\gamma_r + \gamma_i} \} dxdy \tag{3.20}$$

Since the image is divided by indivaidul pixel, x and y value are discrete. So the above integration in the computation for the component of M matrix and D matrix aree performed as summations of each discrete value. Equation (3.19) then becomes

$$M_{ij} = \sum_{i=0}^{I_n - 1} \sum_{j=0}^{J_n - 1} \{ \lambda_{nj} X^{\alpha_j + \alpha_i} Y^{\beta_j + \beta_i} (Q^{-1} - D)^{-\gamma_j + \gamma_i} \} \tag{3.21}$$

and Equation (3.20) becomes

$$D_i = \sum_{i=0}^{I_n - 1} \sum_{j=0}^{J_n - 1} \{ \lambda_{nj} X^{\alpha_r + \alpha_i} Y^{\beta_r + \beta_i} (Q^{-1} - D)^{-\gamma_r + \gamma_i} \} \tag{3.22}$$

3.3 Simulation Results

There are two types of stereo images which are generated and used in the simulation.

1. The first type of stereo images is a pair of images. The left image is generated according to Equation (3.4), the right is also generated according to Equation (3.4) with $\phi = \phi + \psi$ as discussed in Section 3.1.

2. Another type of stereo images is a pair of the Composite Images (see page 31), i.e., each image is formed from a set of images.

3.3.1 Single Pair of Images

By single pair of images, we mean the first type of stereo images just discussed. The following results listed in Table 3.2 were obtained in simulation with the stereo images generated when the values of k_1, k_2 and k_3 are 4096.0, 0.0 and 0.75,

Table 3.2 Simulation results for k1 = 4096

n	simulation value for λ_n	actual value for λ_n
λ_0	−15.999	−16.000
λ_1	0.091	0.000
λ_2	−0.041	0.000
λ_3	−0.049	0.000
λ_4	1.000	1.000
λ_5	0.004	0.000
λ_6	0.034	0.000
λ_7	0.097	1.000
λ_8	0.022	0.000
λ_9	1.014	1.000

respectively. The size of the image is 128 x 128 pixels. The rotation angle of the right camera ψ is equal to 1.475 degree(see Figure 3.1)

If we only change the value of k_1 and keep the values $k_2 = 0.0$, $k_3 = 0.75$, we will get the Table 3.3:

Table 3.3 λ versus k_1

k_1	λ_0	λ_4	λ_7	λ_9
16.0	-7.174	1.000	0.503	0.204
32.0	-4.984	1.000	0.452	0.268
64.0	-17.031	1.000	0.736	-0.374
128	-17.031	1.000	0.810	0.25
256.0	-16.572	1.000	0.939	0.618
512.0	-16.162	1.000	0.962	0.829
1024.0	-15.846	1.000	0.991	1.024
2048.0	-15.993	1.000	0.995	1.014
4096	-16.010	1.000	0.997	1.013
8192	-16.006	1.000	0.998	1.022
actual value	-16.000	1.000	1.000	1.000

The other coefficients which are not listed on the Table 3.3 tend to approach to 0.0 when the k_1 increases. From Table 3.3 it is found that the simulation results become better, when the k_1 is greater than or equal to 1024. When k_1 is equal to 1024, the equivalent gray level range is from 0 to 2047. Also it is found that when the k1 increases the simulation results tend to approach the actual values. The reason is that when the k1 increases, the gray level range widens, and the relative quantization error is decreased.

3.3.2 The Pair of Composite Images

Definition: a Composite Image is an image which is constructed by accumulating the corresponding pixel values of a given set of images of the same object. The following description will give more details about how the construction of the Composite Image is implemented.

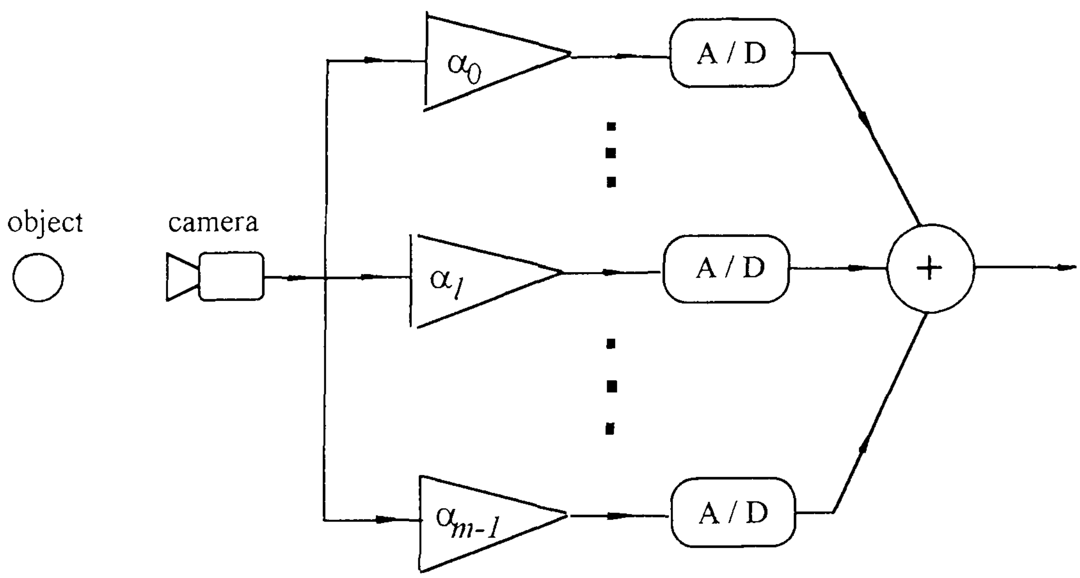

Figure 3.3 The construction of Composite Images

As shown in Figure 3.3, a camera takes an image of an object, and in this image there is a pixel with an analogy gray value $A[i][j]$, where the parameters i and j describe the location of the pixel. When this image signal passes through the lth amplifier, (the gain ratio of the lth amplifier is α_l), the pixel gray value will become $A_l[i][j]$ in the lth *derived image*. We then have:

$$A_l[i][j] = \alpha_l * A[i][j]. \tag{3.22}$$

where α_l is smaller than 1.0, but close to 1.0.

After passing through an 8-bit A-D converter, $A_l[i][j]$ is quantized and becomes an integer, $g_l[i][j]$. Then the pixel gray value of the Composite Image G[i][j] is the sum of the corresponding pixel values in all the derived images.

$$G[i][j] = \sum_{l=0}^{m-1} g_l[i][j] = \sum_{l=0}^{m-1} \text{int}(A_l[i][j]) = \sum_{l=0}^{m-1} \text{int}(\alpha_l * A[i][j]) \tag{3.23}$$

where m is the total number of the derived images. The value of m depends on what kind of image one wants to simulate.

$$m = \frac{2^{n-1}}{128} \tag{3.24}$$

where n is the number of bits which we want to use to represent the gray level of one pixel. If you want to compose an image of which the quantization resolution is 13 bits, the m is equal to 32.

In the simulation with a single pair of stereo images, which is generated by Equation (3.4), it is found that the simulation result will become acceptable when k_1 is greater than or equal to 1024. When k_1 is equal to 1024, the equivalent gray value range is from 0 to 2047. The gray level in real images taken by general CCD camera systems usually ranges from 0 to 255, since most of CCD camera are 8-bit in quantization resolution. The gray level range is not wide enough for this direct method in recovering the surface structure. Hence the Composite Image could be used. Since each pixel gray value G[i][j] in the Composite Image is the sum of the corresponding pixel gray value g_l[i][j] in the derived image, the gray value range in the Composite Image is expanded.

In the simulation we assume that the amplifier gain rate α_l is (128-i)/128. The simulation uses the stereo images which are constructed by m pairs of derived images. The original image A[i][j] is generated by Equation (3.4), where k_1, k_2, k_3 are assigned with 128, 0.0, 0.75, respectively. The rotation angle is 1.475 degree.

Table 3.4 Simulation results with *m* Pairs of Images

m pairs of images	λ[0]	λ[4]	λ[7]	λ[9]	gray level range
1	-16.477	1.000	0.789	0.295	0 ~ 255
2	-16.522	1.000	0.856	0.464	0 ~ 511
4	-16.137	1.000	0.923	0.754	0 ~ 1011
8	-16.052	1.000	0.959	0.881	0 ~ 2023
16	-15.962	1.000	0.976	0.954	0 ~ 4095
32	-15.954	1.000	0.990	0.975	0 ~ 8091

From Table 3.4, it is found that when the number of derived images is increased, the gray level range is extended, and the simulation results tend to the actual values.

In Table 3.2, the simulation results with a single pair of stereo images, it is found that the simulation result is best when the gray level range is from 0 to 8191 (2^{13}-1). Thus, to get the same gray level range for the Composite Image, the equivalent total number of derived image is 32. Table 3.5 contains the simulation results for the Composite Images with m equal to 32. The original stereo images are created by Equation (3.4), the brightness generate function. The value of k_1, k_2 and k_3 are equal to 128, 0.0 and 0.75, respectively and the rotation angle of right camera ψ is equal to 1.475. Figures 3.4 and 3.5 are the single pair of images. Figures 3.6 and 3.7 are the pair of the Composite Images with each pixel gray value by 28.

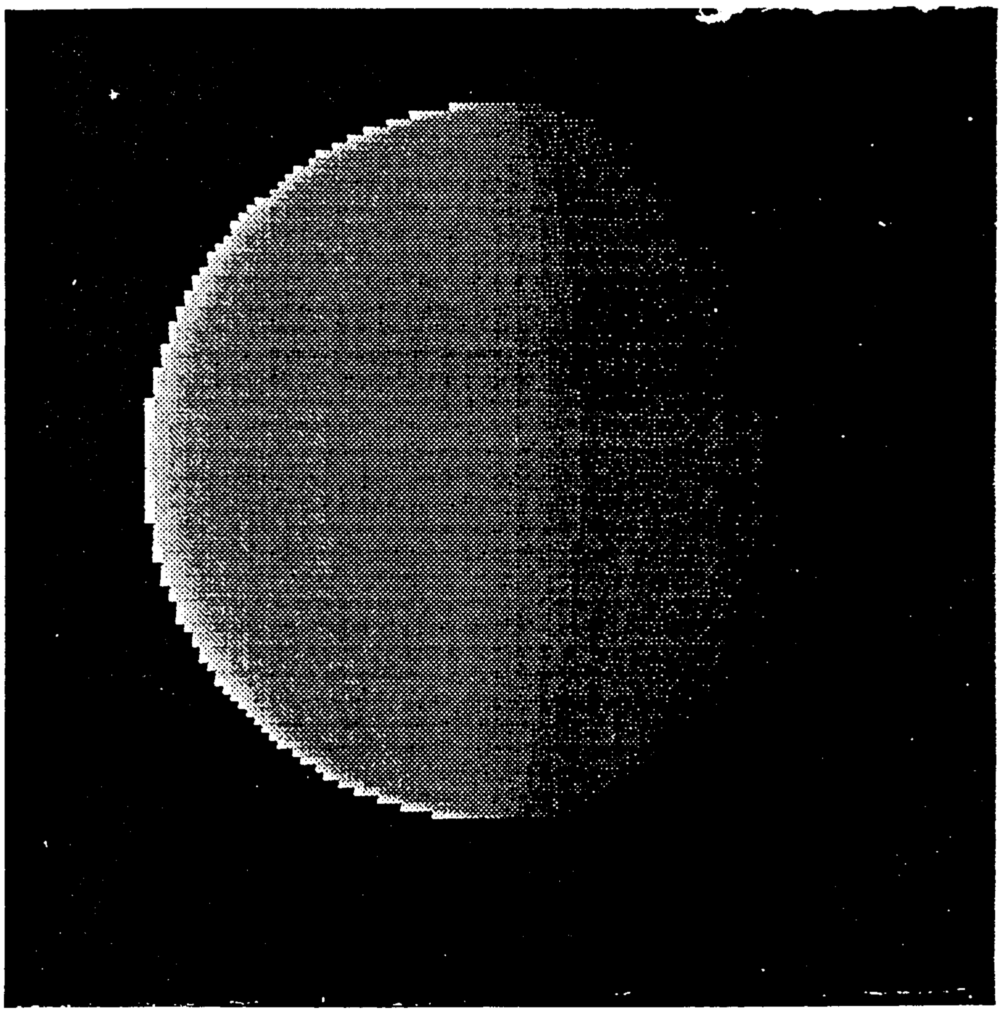

Figure 3.4 The left single image

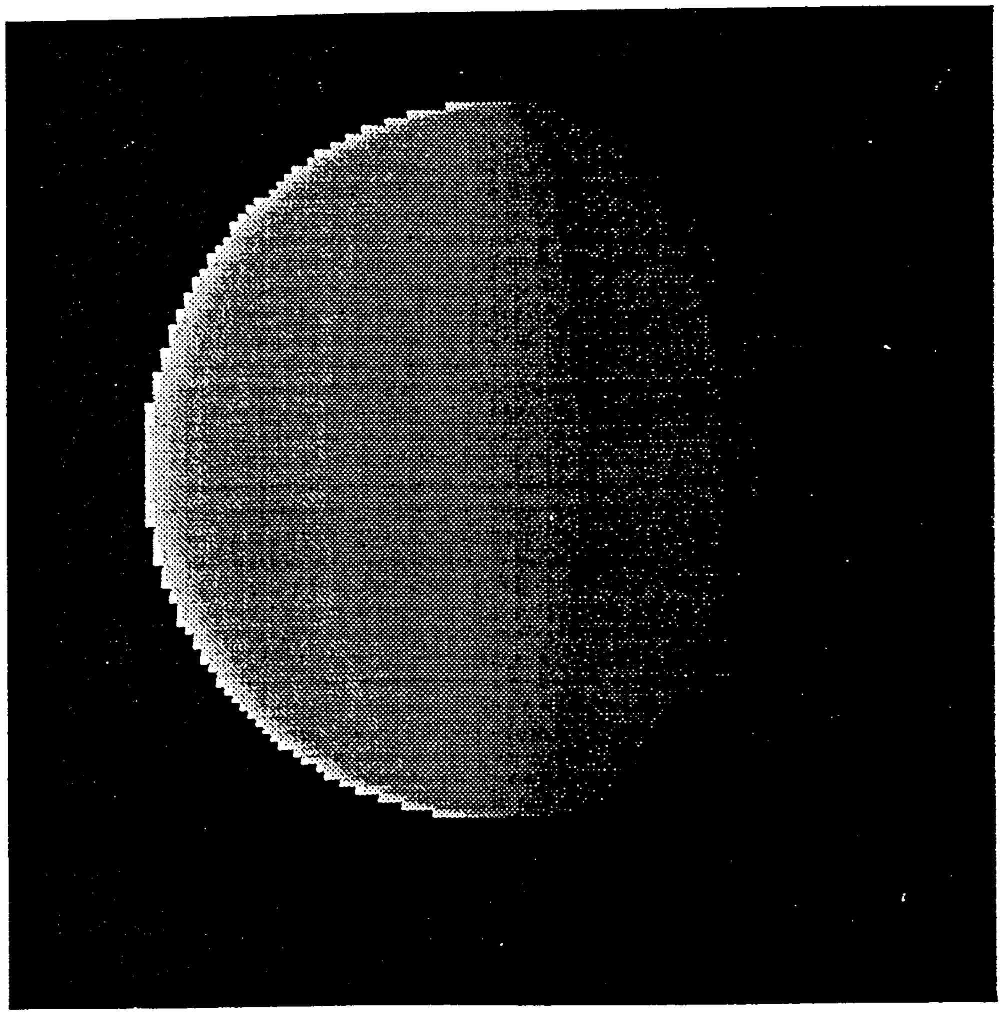

Figure 3.5 The right single image

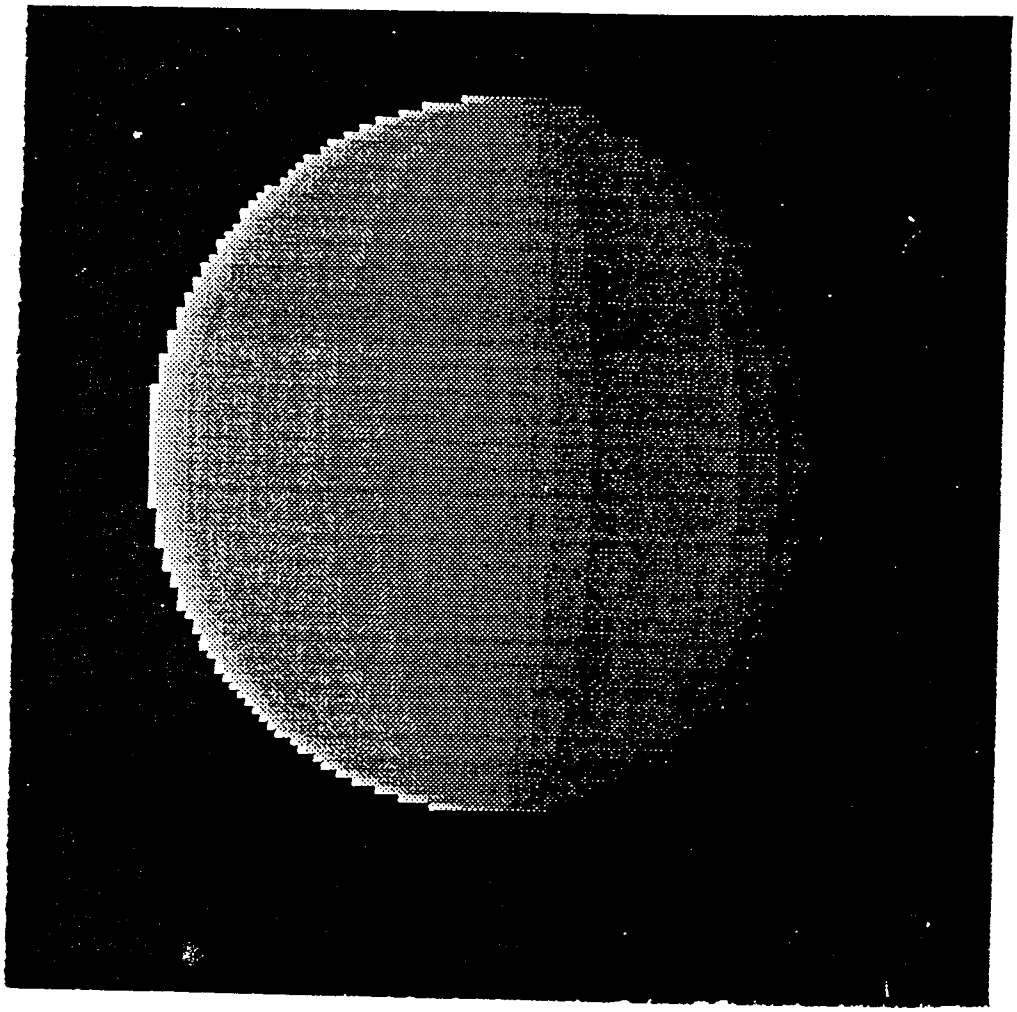

Figure 3.6 The left Composite Image

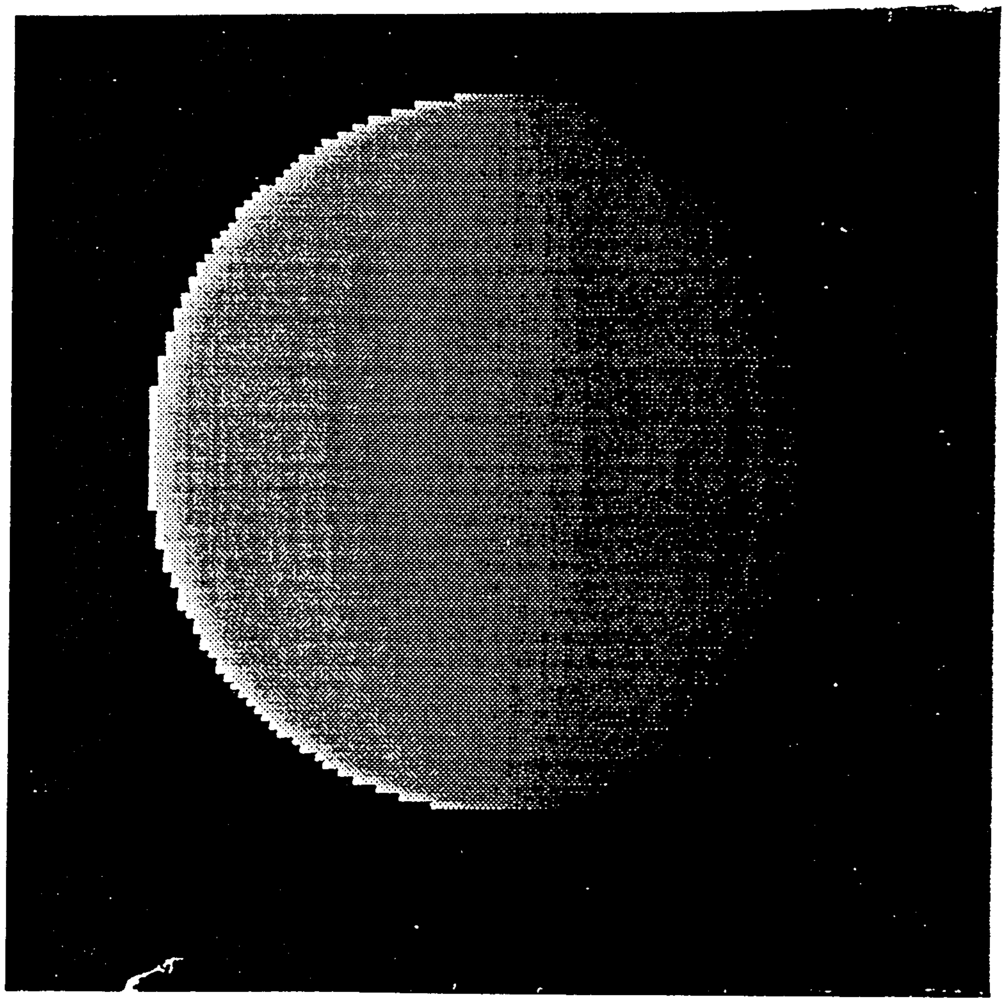

Figure 3.7 The right Composite Image

Table 3.5 Simulation results: the value of $\lambda[i]$,while $k_2 = 0.0$, $k_3 = 0.75$, ψ=1.475, α_l=(128-i)/128.

$\lambda[i]$	simulation result	ideal value
$\lambda[0]$	-15.954	-16.000
$\lambda[1]$	0.033	0.0
$\lambda[2]$	-0.070	0.0
$\lambda[3]$	-0.089	0.0
$\lambda[4]$	1.000	1.000
$\lambda[5]$	-0.001	0.0
$\lambda[6]$	-0.007	0.0
$\lambda[7]$	0.0985	1.000
$\lambda[8]$	0.005	0.0
$\lambda[9]$	0.991	1.000

As generally, the amplifier gain ratio α_l will vary while the working conditions change, such as temperatures. So we assume the amplifier gain ratios α_l are randomly changed between 0.8 to 1.0, we will get the simulation results $\lambda[i]$ listed in Table 3.6. The values of α_l are :

$\alpha_0 = 1.000$	$\alpha_1 = 0.9015$	$\alpha_2 = 0.8035$
$\alpha_3 = 0.8125$	$\alpha_4 = 0.9304$	$\alpha_5 = 0.9725$
$\alpha_6 = 0.8012.$	$\alpha_7 = 0.8245$	$\alpha_8 = 0.9924$
$\alpha_9 = 0.9823$	$\alpha_{10} = 0.9478$	$\alpha_{11} = 0.9425$
$\alpha_{12} = 0.8834$	$\alpha_{13} = 0.8223$	$\alpha_{14} = 0.8213$
$\alpha_{15} = 0.9224$	$\alpha_{16} = 0.9323$	$\alpha_{17} = 0.9769$
$\alpha_{18} = 0.9832$	$\alpha_{19} = 0.9324$	$\alpha_{20} = 0.9456$
$\alpha_{21} = 0.8786$	$\alpha_{22} = 0.8954$	$\alpha_{23} = 0.9324$

$\alpha_{24} = 0.9767$ $\alpha_{25} = 0.9883$ $\alpha_{26} = 0.9546$

$\alpha_{27} = 0.9564$ $\alpha_{28} = 0.9534$ $\alpha_{29} = 0.9634$

$\alpha_{30} = 0.8936$ $\alpha_{31} = 0.9783$

Table 3.6 Simulation results: the value of $\lambda[i]$, while $k_2 = 0.0$, $k_3 = 0.75$, ψ=1.475 degree, α_l is between 0.8 to 1.0.

$\lambda[i]$	simulation result	ideal value
$\lambda[0]$	-16.040	-16.000
$\lambda[1]$	0.068	0.0
$\lambda[2]$	-0.077	0.0
$\lambda[3]$	-0.188	0.0
$\lambda[4]$	1.000	1.000
$\lambda[5]$	-0.010	0.0
$\lambda[6]$	-0.002	0.0
$\lambda[7]$	0.0974	1.000
$\lambda[8]$	0.002	0.0
$\lambda[9]$	0.959	1.000

3.3.3 Analysis

(1) The Composite Image method can extend the gray level.

From Table 3.4, it is found that the gray level range is extended when the number of the derived images is increased. It can be explained by Equation (3.23). Each pixel gray value in the Composite Image is the sum of the corresponding pixel values in the derived images. So when the number of derived images increases, the summation will increase.

(2) The Composite Image method can make the first order of derivatives more precise.

From Equations (2.29), (2.35) , it is found that the recovered surface heavily depends on the first order derivative of the pixel. In the Composite Image, the first order derivatives in The Composite Image will be more accurate than that in the original image. Consider two neighboring pixels in the image, their gray values are

A[i][j] = 123.3

A[i-1][j] = 124.8

where the gray level range is 0 to 255.

In the single image, we get the actual first order derivative as

first order derivative = A[i][j] - A[i-1][j]

= 123.3 - 124.8

= -1.5

where the distance between two neighboring pixels is assumed to be one unit. Since the gray value retrieved from the A/D convert is integer, the first order derivative then becomes

first order derivative = int(A[i][j]) - int(A[i-1][j])

= 123 - 125

= -2

Obviously the quantization error is -0.5, which is the difference between the above two values.

In the Composite Image, the total number of derived images, m, is 32. The amplifier gain ratio α_l is given by $(128- l)/128$, l is from 0 to 31. Using Equation (3.23),

$$G[i][j] = \sum_{l=0}^{m-1} g_l[i][j] = \sum_{l=0}^{m-1} \mathrm{int}(A_i[i][j]) = \sum_{l=0}^{m-1} \mathrm{int}(\alpha_l * A[i][j]) \qquad (3.23)$$

we will be able to get these two gray values G[i-1][j], G[i][j] in the Composite Image, which are corresponding to A[i][j-1], A[i][j] in the single image

G[i][j]	= 3466
G[i-1][j]	= 3508

The first order derivative in the Composite Image is

first order derivative = G[i][j] - G[i-1][j]

= 3466 - 3508

= -42

The gray value is expanded by almost 28 times. If the gray value is transferred to the original gray value range, the first order derivative is -1.48 (-42 divide 28.12), which is very close to -1.5. So the first order derivative in the Composite Image will be more close to the actual value. Table 3.7 has shown why the first order derivative will be more close to the real value. The first column in Table 3.7 contains the indices. The second column contains the values of α_l. The third and fourth contain quantization values for A[i][j]$*\alpha_l$ and A[i-1][j]$*\alpha_l$, respectively. These are the quantization gray values in the lth derived image. The fifth is the first order derivative in lth derived image, which is the difference between the item in the third and fourth columns, respectively. The sixth column is the quantization error. From Table 3.7, it is found that in each derived image the first order of derivative sometimes will be bigger than the actual value of the first order of derivative and causes the positive quantization error. Sometimes it is smaller than the actual value of first order derivative and causes negative error. The positive error and negative error will counteract each other, and the first order of derivative will tend to approach the actual value. For example when l index is 6, the value of α_l is 0.953. By equation (3.23), two pixels float gray values are 117.50 and 118.93, respectively. The quantization values are 118 and 119. So the first order of derivative with float value is -1.43, and the first order of derivative with quantization value is -1.00. The quantization error is -0.43. When index l is 5, the value of α_l is 0.961. The two pixels float gray values are 118.49 and 119.93, respectively. The quantization values are 118 and 120. So the first order of derivative with float value is -1.44, and the first order of derivative with quantization value is -2.00. The quantization error is 0.56. If we add these two derived images, the quantization error is 0.13. It is very close to zero. That is why the Composite Images will have a high precision in first order derivative.

Table 3.8 has shown the first order of derivative for another pair of pixels, which gray values are 124.4 and 124.3. Although the difference between two pixels gray values are very small. However the similar result has been achieved, the first order derivatives in the Composite Image will be more precise than that in the original image. The actual first order derivative in the original image is -0.1 (124.4 - 124.3). The first order derivative for the quantization value is 0.0 (124 - 124). So the quantization error is 0.1, which is equal to the actual value minus the actual value. The gray values in the Composite Image are 3500 and 3497 respectively. So the first derivative is 3, and transferred to original gray value range, the first derivative is equal to 0.11(3.0 divide 28.12).

Up to now we know by using the Composite Image one can extend the range of gray level, and enhance the precision of the first order derivatives.

Table 3.7 The first order derivatives for A[i-1][j] = 123.3, A[i][j] = 124.8.

index	α_l	int(α_l*A[i-1][j])	int(α_l*A[i][j])	first order derivative	quantization error
0	1.000	123	125	-2.000	0.500
1	0.992	122	124	-2.000	0.512
2	0.984	121	123	-2.000	0.523
3	0.977	121	122	-2.000	0.535
4	0.969	120	121	-2.000	0.547
5	0.961	119	120	-2.000	0.559
6	0.953	118	119	-1.000	-0.430
7	0.945	117	118	-1.000	-0.418
8	0.938	116	117	-1.000	-0.406
9	0.930	115	116	-1.000	-0.395
10	0.922	114	115	-1.000	-0.383
11	0.914	113	114	-1.000	-0.371
12	0.906	112	113	-1.000	-0.359
13	0.898	111	112	-1.000	-0.348
14	0.891	110	111	-1.000	-0.336
15	0.883	109	110	-1.000	-0.324
16	0.875	108	109	-1.000	-0.312
17	0.867	107	108	-1.000	-0.301
18	0.859	106	107	-1.000	-0.289
19	0.852	105	106	-1.000	-0.277
20	0.844	104	105	-1.000	-0.266
21	0.836	103	100	-1.000	-0.254
22	0.828	102	103	-1.000	-0.242
23	0.820	101	102	-1.000	-0.230
24	0.812	100	101	-1.000	-0.219
25	0.805	99	100	-1.000	-0.207
26	0.797	98	99	-1.000	-0.195
27	0.789	97	98	-1.000	-0.184
28	0.781	96	98	-2.000	0.828
29	0.773	95	97	-2.000	0.840
30	0.766	94	96	-2.000	0.852
31	0.758	93	95	-2.000	0.863
total	28.12	3466	3508	-42	-0.185

Table 3.8 The first order derivatives for A[i-1][j] = 124.4, A[i][j] = 124.3.

index	α_l	int(α_l*A[i-1][j])	int(α_l*A[i][j])	first order derivative	quantization error
0	1.000	124	124	0.000	0.100
1	0.992	123	123	0.000	0.099
2	0.984	122	122	0.000	0.098
3	0.977	121	121	0.000	0.098
4	0.969	121	120	1.000	-0.903
5	0.961	120	119	1.000	-0.904
6	0.953	119	118	1.000	-0.905
7	0.945	118	118	0.000	0.095
8	0.938	117	117	0.000	0.094
9	0.930	116	116	0.000	0.093
10	0.922	115	115	0.000	0.092
11	0.914	114	114	0.000	0.091
12	0.906	113	113	0.000	0.091
13	0.898	112	112	0.000	0.090
14	0.891	111	111	0.000	0.089
15	0.883	110	110	0.000	0.088
16	0.875	109	109	0.000	0.087
17	0.867	108	108	0.000	0.087
18	0.859	107	107	0.000	0.086
19	0.852	106	106	0.000	0.085
20	0.844	105	105	0.000	0.084
21	0.836	104	100	0.000	0.084
22	0.828	103	103	0.000	0.083
23	0.820	102	102	0.000	0.082
24	0.812	101	101	0.000	0.081
25	0.805	100	100	0.000	0.080
26	0.797	99	99	0.000	0.080
27	0.789	98	98	0.000	0.079
28	0.781	97	97	0.000	0.078
29	0.773	96	95	0.000	0.077
30	0.766	95	95	0.000	0.077
31	0.758	94	94	0.000	0.076
total	28.12	3500	3497	3.000	0.482

CHAPTER IV

CONCLUSION

In this thesis, based on the unified optical flow field, a direct method in space domain is used for recovering the surface structure from a pair of stereo images. The direct method does not need to explicitly solve the optical flow field and to find feature correspondence. Compared with the direct method in time domain, which can only estimate the planar surface, the direct method in space domain could recover the surface structure of an object which can be characterized by an Nth degree polynomial equation.

4.1 Observation

Gray Level Range: From the simulation results, it is found that the coefficients in polynomial equation will tend to approach the actual values, when the gray level range becomes wide. The reason is that the direct method to recover the surface structure of objects heavily depends on the precision of the first order derivatives. If the range of gray level is not wide enough, the neighboring pixel could not be distinguished by their gray values since their small difference in their gray values may sometimes vanish into the quantization errors. This will cause significant error in surface recovery. Therefore a wider gray level range is expected to bring about better results. As mentioned before, the Composite Image can extend gray value range, and the simulation results agree with this expectation.

Robustness of Composite Image: In total, 100,000 groups of $\alpha_{l,}$ values are randomly created by using certain computer software. In each group of α_l values there are 32 random numbers, ranging from 0.8 to 1.00. These 100,000 groups of

α_l are used for constructing 100,000 pairs of composite stereo images. After we use these 100,000 pairs of composite stereo images to recover the surface structures, the maximum and minimum values for λ[0], λ[4], λ[7], λ[9], which are coefficients of constant, X^2, Y^2, Z^2 monomials in the 2nd degree of polynomial equation, are obtained. They are listed in Table 4.1.

Table 4.1 Maximum and minimum values for λ[0], λ[4], λ[7], λ[9]

	λ[0]	λ[4]	λ[7]	λ[9]
maximum value	-16.230	1.000	0.9995	0.9981
minimum value	-16.023	1.000	0.9625	0.9316
actual value	-16.000	1.000	1.000	1.000

The simulation results in Table 4.1 are quite good. Hence the Composite Image method could be considered robust.

4.2 Accomplishment

Simulation results have shown that in order to obtain the results which are close to actual value the gray level range needs to be wider than 2048. However the gray level in real images taken by general CCD camera systems (which are 8-bit in quantization resolution.) usually ranges from 0 to 255. Therefore, the gray level range in the real images is too narrow to use this direct method to recover the surface structure. In contrast to single pair of images, the gray level range is extended with the Composite Image, it makes it possible to build a system that is described in this thesis (shown in Figure 3.3), with the current technology in the solid state industry to recover surface structures from image sequences.

WORKS CITED

1. Aggarwel, J. K., and N. Nandhakumar. "On the Computation of Motion from Sequences of Images - A Review." *Proceedings of the IEEE, Vol. 76, No.8,* August (1988): *917-935.*

2. Aloimonos, J., and J. Herve. "Correspondenceless Stereo and Motion: Planar Surfaces." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.12, No.5, May (1990): 504-510.

3. Hayashi, B., and S. Negahdaripour. "Direct Motion stereo." Proceedings of SPIE, Vol.1260, Sensing and Reconstruction of Three-Dimensional Objects and Scenes, February (1990): 78-85.

4. Heel, J., and S. Negahdaripour. "Time-sequential Structure and Motion Estimation without Optical Flow," *Proceedings of SPIE, Vol.1260, Sensing and Reconstruction of Three-dimensional objects and Scenes,* February (1990): *50-61.*

5. Horn, B. K. P., and B. G. Schunck. "Determining optical flow." *Artificial Intelligence* (1981): 185-203.

6. Horn, B. K. P., and E. J. Weldon Jr. "Direct Methods for Recovering Motion." *Journal of Computer Vistion,* No.2 (1988): 51-76.

7. Negahdaripour, S., and B. K. P. Horn. "Direct passive navigation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.PAMI-9, No.8 January (1987): 168-176.

8. Shi, Y. Q., Y. Zhu, and M. A. Salhi. "Direct Recovering of Nth order surface structure, using the UOFF Approach." *Proceedings of IEEE 1992 International Symposium on Circuits and System.* May (1992): 1475-1478.

9. Shu, C. Q., and Y. Q. Shi. "Computation of Motion from Stereo Image sequence using Unified Optical Flow Field." presented at *SPIE's 1990 International Symposium on Optical and Optoelectronic Applied Science and Engineering*, San Diego, CA, July (1990) and collected in Applications of Digital Image Processing XIII, andrew G. Tescher, Editior, Proc. SPIE 1349 (1990): 346 - 357.

10. Shu, C. Q., and Y. Q. Shi. "On Unified Optical Flow Field." *Pattern Recognition*, vol. 24, no. 6. June (1991): 579-586.

11. Shu, C. Q., Y. Zhu, Y. Q. Shi, and C. H. Lu. "Recovering Surface Structure Characterized by an Nth Degree Polynomial Equation." *IEEE Seventh Workshop on Multidimension Signal Processing*, 5.7, September (1991).

12. Thompson, W. B. "Introduction to the spacial issue on visual motion." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.11, No. 5, May (1989): 449-450.

13. Weng, J., T. S. Huang, and N. Ahuja. "Motion and structure from two perspective views: algorithum, error analysis and error estimation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.11, No. 5, May (1989): 451-476.