1-31-1993

Optical flow determination and motion analysis

Qian Wang
New Jersey Institute of Technology

ABSTRACT

Optical Flow Determination and Motion Analysis

by

Qian Wang

Optical flow, also known as image flow, is of fundamental importance in the processing of image sequences. Various approaches to determine optical flow have been proposed --- others continue to appear. Using several sets of real image sequences, here, the results of three different approaches to the the determination of optical flow are reported, the approaches are the Gradient-based approach, the Correlation-based approach and the Correlation-feedback approach. The comparisons are primarily empirical, and they show that the performance of these different algorithms differs significantly among the different image sequences, indicating that there does not exists one superior algorithm which could be suitable to every different kind of situations. Every individual algorithm has its own advantages and limitations.

Also, recent research shows that motion analysis has many potentials in the computer vision area. There are basically two different approaches to recover structure of objects and relative motion between objects and cameras: one is the optical flow field approach and the other is the feature correspondence approach. The unified optical flow field (UOFF) [5] is a generalization of the optical flow to stereo imagery. Here, through the results obtained from the real image sequences, the feasibility of UOFF approach to motion analysis has been shown.

OPTICAL FLOW DETERMINATION AND MOTION ANALYSIS

by
Qian Wang

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

Optical Flow Determination and Motion Analysis

Qian Wang

[illegible] 12/15/92

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

[illegible] 12/15/92

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

[illegible] 12/15/92

Dr. Marshall Kuo, Committee Member
Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Qian Wang

Degree: Master of Science in Electrical and Computer Engineering

Date: January, 1993

Date of Birth:

Place of Birth:

Undergraduate and Graduate Educations:

- Master of Science in Electrical and computer Engineering, New Jersey Institute of Technology, Newark, NJ, 1993
- Bachelor of Engineering in Electrical Engineering, Huazhong University of Science and Technology, Wuhan, P. R. China, 1988

Major: Electrical and Computer Engineering

This thesis is dedicated to
my dearest parents Guoqi Wang, Hui Yi
and my dearest brothers and sisters
Xuanshi Wang, Ying Liang and Dong Liu

ACKNOWLEDGMENT

I would like to take this opportunity to acknowledge and thank Dr. Yun-Qing Shi, my thesis advisor, for his precious guidance and expert assistance during the research and writing for this thesis. I would also like to express my sincere appreciation and gratitude to Dong Liu and Jingning Pan, for their constructive advice and support to successfully complete this thesis. Special thanks go to Dr. Marshall Kou and Dr. Edwin Hou, for being members of this thesis committee, and reading and evaluating this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Optical flow, also known as image flow, is a very important property of image field. Optical flow means assigning velocities to each point in the 2-dimension image field which is the perspective projection of the 3-dimension object's surface points. The 2-d velocity vector depicts the projection of the instantaneous 3-d velocity of the corresponding point of the object. The information of the motion of a 3-d object in a dynamic sense is contained in an image sequence, therefore the image flow should be computed from the image sequence. However, it cannot be computed locally because only one independent measurement is available from an image sequence at a given point, while two components of velocity are required to be computed successfully. This is the major challenge in calculation of optical flow.

Various methods have been proposed for computing optical flow, and we can expect more methods to be introduced in the near future. Most of the current approaches to the computation of optical flow can be divided into different categories according to their selection of measurements.

A study and comparison of the following three different optical flow algorithms are reported in this thesis: Gradient-based, Correlation-based and Correlation-feedback approaches, represented by Horn and Schunck, Singh and Pan's algorithm respectively. Despite their differences in implementation, all these three algorithms can be viewed conceptually similar in terms of going through the following procedure: Extracting the interested signal structure from the real image sequence, enhancing the signal-to-noise ratio of it, selecting the basic measurements, then combining these measurements to create a 2-d optical flow field with the smoothness assumption of the underlying optical flow field.

In this thesis work, after comparing the results calculated with the above algorithms in three different settings, the conclusion is that no single algorithm is most suitable to calculate optical flow at all the different kind of situations. Every individual algorithm has its own advantages and limitations. We should use different algorithms to calculate the optical flow in different cases. The results obtained from real image sequences proved this observation.

Over the last ten years, motion analysis has become a very active research subject in the computer vision area. In the vision system, recovery of motion of objects gives us the ability to estimate their speed and direction and then to navigate, recognize and track objects. Optical flow field approach and feature correspondence approach are the two major different approaches to recovery structure of objects and relative motion between objects and cameras.

The feature correspondence approach requires the solution to feature detection and correspondence establishment, which have been proven to be computationally and conceptually difficult to solve. And only partial solutions suitable for simplistic situation have been developed.

The optical flow field approach suggests that we should use the distribution of apparent velocities of movement of image brightness patterns over a large portion of the image, to recover the object's position and motion. These methods achieve more robustness at the expense of more computation which is encountered mainly in calculation of the optical flow field. Relatively, the analysis of the motion of objects is less computational intensive. However optical flow cannot be computed locally, it must be computed under some constraint, so in many cases an estimated optical flow field is not the same of the true motion field.

Recently, a new approach to motion analysis from a sequence of stereo images has been developed. Feature correspondence is not necessary in this approach, so it is more efficient. This new approach is based on an unified optical flow field (UOFF), combining both the spatial and the temporal domains. The recovered 3-d motion is valid for a whole continuous moving field instead of only valid for some features.

In this thesis, the new concept of UOFF is studied , and two different sets of equations which are corresponding to two different imaging geometries to recovery the position and motion of 3-d objects have been derived.

CHAPTER 2

IMAGE FLOW DETERMINATION

2.1 Three Different Optical Flow Approaches

The recovery of 3-d motion and structure from corresponding image sequences can be archived by two steps: 1) Compute image velocities from the changes of image intensity value, i.e., the optical flow. 2) Compute 3-d motion and structure using optical flow as an intermediate result. The first step is the most difficult part. So far, there does not exist one unique technique that can cover all situation.

In this thesis, three different approaches, gradient-based approach, correlation-based approach and correlation-feedback approach have been studied.

2.1.1 Gradient-based Approach

Horn and Schunck's technique is a typical representative of the gradient-based approach [1]. It is a differential technique which computes the velocity from spatiotemporal derivatives of image intensity.

Let $I(x, y, t)$ denote the image brightness at the point(x, y) in the image plane at time t. Assume the brightness of a particular point in the pattern is constant when the pattern moves, that is,

$$\frac{dI}{dt} = 0 \tag{2.1}$$

which can also be written as

$$I(x, y, t) \approx I(x + \delta x, y + \delta y, t + \delta t) \tag{2.2}$$

we call the above equation as the intensity-constancy assumption.

Expand the right side into Taylor's series in the vicinity of point (x, y, t), we have

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + \varepsilon \qquad (2.3)$$

where ε contains the second and higher order terms of δx,δy and δt.

From equations (2.2) and (2.3), we can get

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + \varepsilon = 0 \qquad (2.4)$$

Dividing both sides by δt, we get

$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t} + \frac{\varepsilon}{\delta t} = 0 \qquad (2.5)$$

Considering the fact that ε contains the second and higher order terms of δx,δy and δt, we can ignore the term $\varepsilon/\delta t$ when $\delta t \to 0$, this gives us

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \qquad (2.6)$$

If we let $u \triangleq dx/dt$, $v \triangleq dy/dt$, we can get

$$I_x u + I_y v + I_t = 0 \qquad (2.7)$$

where $I_x \triangleq \frac{\partial I(x,y,t)}{\partial x}$,$I_y \triangleq \frac{\partial I(x,y,t)}{\partial y}$, $I_t \triangleq \frac{\partial I(x,y,t)}{\partial t}$ are the partial derivatives of $I(x, y, t)$ with respect to x, y, t, respectively.

It is not sufficient to compute the optical flow by just using the above equation. To solve this problem, we usually add the smoothness constraint as an auxiliary equation which is minimize the square of the magnitude of the gradient of the optical flow velocity:

$$(\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 \quad and \quad (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2$$

In practice, the error comes from two major sources, one is the rate of the change of image brightness, and the other is the departure from smoothness in the image velocity, these can be expressed as the following equations:

$$\varepsilon_b = I_x u + I_y v + I_t \tag{2.8}$$

$$\varepsilon_c^2 = (\frac{\delta u}{\delta x})^2 + (\frac{\delta u}{\delta y})^2 + (\frac{\delta v}{\delta x})^2 + (\frac{\delta v}{\delta y})^2 \tag{2.9}$$

The problem to calculate the optical flow becomes to minimize the sum of the error of the above two sources. For some practical reasons, we should choose a suitable weighting factor, denoted by α^2. Now, we can compute the optical flow by minimizing the following error equation.

$$\varepsilon^2 = \int\int (\alpha^2 \varepsilon_c^2 + \varepsilon_b^2)\, dx\, dy \tag{2.10}$$

By the calculus of variation, we obtain

$$I_x^2 u + I_x I_y v = \alpha^2 \nabla^2 u - I_x I_t \tag{2.11}$$

$$I_x I_y u + I_y^2 v = \alpha^2 \nabla^2 u - I_y I_t \tag{2.12}$$

Using the approximation of the Laplacian,

$$\nabla^2 u \approx k(\overline{u} - u)$$

$$\nabla^2 v \approx k(\overline{v} - v)$$

In this situation, we can assume $k = 1$. By using the above four equations, we can solve Equation (2.10) in the following way:

$$(\alpha^2 + I_x^2)u + I_x I_y v = (\alpha^2 \overline{u} - I_x I_t) \tag{2.13}$$

$$I_x I_y u + (\alpha^2 + I_y^2)v = (\alpha^2 \overline{v} - I_y I_t) \tag{2.14}$$

The determinant of the coefficient matrix is $\alpha^2(\alpha^2 + I_x^2 + I_y^2)$. Solving these equations with respect to u and v we find that

$$(\alpha^2 + I_x^2 + I_y^2)u = +(\alpha^2 + I_y^2)\overline{u} - I_x I_y \overline{v} - I_x I_t \tag{2.15}$$

$$(\alpha^2 + I_x^2 + I_y^2)v = -I_x I_y \overline{u} + (\alpha^2 + I_x^2)\overline{v} - I_y I_t \tag{2.16}$$

Now, there is a pair of equations for every point in the image. But it would be very costly to solve these equations simultaneously by some of the standard method, such as Gauss-Jordan elimination. So iterative equations are used to minimize the equation 2.10, and image velocity (u, v) can be obtained as

$$u^{n+1} = \overline{u}^n - I_x \frac{I_x \overline{u}^n + I_y \overline{v}^n + I_t}{(\alpha^2 + I_x^2 + I_y^2)} \tag{2.17}$$

$$v^{n+1} = \overline{v}^n - I_y \frac{I_x \overline{u}^n + I_y \overline{v}^n + I_t}{(\alpha^2 + I_x^2 + I_y^2)} \tag{2.18}$$

where u^0 and v^0 are the initial velocity estimations and are set to zero everywhere, $\overline{u}$ and $\overline{v}$ denote the neighborhood averages of image velocity.

2.1.2 Correlation-based Approach

Let's use Singh's technique as a typical representative of the correlation-based approach. Singh's method is a region-based matching technique, which computes the velocity as the shift d that yields the best fit between image regions at different time. Finding the best match is equivalent to maximizing a similarity measure (over d), such as the normalized cross-correlation, or minimizing a distance measure, such as the sum-of-squared difference (SSD) [4].

$$\varepsilon_c(d_x, d_y) = \sum_{j=-n}^{n} \sum_{i=-n}^{n} w(i,j)[I_1(x+i, y+j) - I_2(x+d_x+i, y+d_y+j)]^2 \tag{2.19}$$

where w denotes a 2-d window function, and (d_x, d_y) are usually restricted to be small integer numbers.

Singh specifies the matching technique into two steps, in the first step conservation information is extracted, then in the second step neighborhood information is extracted. Finally the image flow can be computed by combining the two estimates on the basis of their covariance-matrices.

Conservation information is extracted based on conservation property of intensity over time. Based on the assumption of conservation, estimating image flow amounts to an explicit search for the best match in a search-area in the subsequent images of a sequence. Correlation gives a response at each pixel in the search window in the subsequent images.

The procedure of extracting the conservation information can be described as, first creating a correlation-window W_p of size $(2n+1)*(2n+1)$ around the pixel at location (x,y) in the first image I_1, then forming the search-window W_s of $(2N+1)*(2N+1)$ around the pixel at location (x,y) in the second image I_2, finally $(2N+1)^2$ samples of error-distribution are computed as following using sum-of-squared-differences as:

$$\varepsilon_c(u,v) = \sum_{j=-n}^{n}\sum_{i=-n}^{n}[I_1(x+i,y+j) - I_2(x+u+i,y+v+j)]^2 \qquad (2.20)$$

$$-N \leq u,v \leq +N$$

The $(2N+1)^2$ samples of response-distribution are computed as an exponential function of error-distribution as follows:

$$R_c(u,v) = e^{-k\varepsilon_c(u,v)} \qquad (2.21)$$

Each point in the search area is a candidate for the ‘‘true match’’. The response at the point can be thought as a weight that reflects the faith in the measurement. Based on the weighted-least-squares estimation, the estimate of image velocity, denoted by $U_{cc} = (u_{cc}, v_{cc})$, can be computed as:

$$u_{cc} = \frac{\sum_u \sum_v R_c(u,v)u}{\sum_u \sum_v R_c(u,v)} \qquad (2.22)$$

$$v_{cc} = \frac{\sum_u \sum_v R_c(u,v)v}{\sum_u \sum_v R_c(u,v)} \qquad (2.23)$$

and the associated covariance-matrix is given by

$$S_{cc} = \begin{pmatrix} \frac{\sum_u \sum_v R_c(u,v)(u-u_{cc})^2}{\sum_u \sum_v R_c(u,v)} & \frac{\sum_u \sum_v R_c(u,v)(u-u_{cc})(v-v_{cc})}{\sum_u \sum_v R_c(u,v)} \\ \frac{\sum_u \sum_v R_c(u,v)(u-u_{cc})(v-v_{cc})}{\sum_u \sum_v R_c(u,v)} & \frac{\sum_u \sum_v R_c(u,v)(v-v_{cc})^2}{\sum_u \sum_v R_c(u,v)} \end{pmatrix} \qquad (2.24)$$

where the summation is carried out over $-N \leq u, v \leq +N$.

Neighborhood information is utilized to propagate velocity. Forming the neighborhood window of size $(2w + 1) * (2w + 1)$, the velocities of these $(2w + 1)^2$ pixels are mapped to the points (u_i, v_i) in $u - v$ space (where $1 \leq i \leq (2w + 1)^2$), with the weight to the point (u_i, v_i) as $R_n(u_i, v_i)$. Based on the weighted-least-squares estimation, the estimate of image velocity of the central pixel denoted by $\overline{U} = (\overline{u}, \overline{v})$, is given by

$$\overline{u} = \frac{\sum_u \sum_v R_n(u, v)u}{\sum_u \sum_v R_n(u, v)} \tag{2.25}$$

$$\overline{v} = \frac{\sum_u \sum_v R_n(u, v)v}{\sum_u \sum_v R_n(u, v)} \tag{2.26}$$

The covariance matrix is

$$S_n = \begin{pmatrix} \frac{\sum_i R_n(u_i,v_i)(u_i-\overline{u})^2}{\sum_i R_n(u_i,v_i)} & \frac{\sum_i R_n(u_i,v_i)(u_i-\overline{u})(v_i-\overline{v})}{\sum_i R_n(u_i,v_i)} \\ \frac{\sum_i R_n(u_i,v_i)(u_i-\overline{u})(v_i-\overline{v})}{\sum_i R_n(u_i,v_i)} & \frac{\sum_i R_n(u_i,v_i)(v_i-\overline{v})^2}{\sum_i R_n(u_i,v_i)} \end{pmatrix} \tag{2.27}$$

where the summation is carried out over $1 \leq i \leq (2w + 1)^2$, $-N \leq u, v \leq +N$.

By estimation theory, the neighborhood error in a quadratic form is

$$(U - \overline{U})^T S_n^{-1} (U - \overline{U})$$

conservation error is the following quadratic form:

$$(U - U_{cc})^T S_{cc}^{-1} (U - U_{cc})$$

where U is used to denote the velocity estimate (u, v)

The sum of conservation error and neighborhood error represents the squared error in the velocity estimate U. The optimal estimate of velocity is the estimate which minimizes the mean squared error over the visual field. That is

$$Minimize \int\int [(U - \overline{U})^T S_n^{-1} (U - \overline{U}) + (U - U_{cc})^T S_{cc}^{-1} (U - U_{cc})] dxdy \tag{2.28}$$

Calculus of variations can be used to solve the above problem,that gives

$$S_{cc}^{-1}(U-U_{cc})+S_n^{-1}(U-\overline{U})=0 \tag{2.29}$$

$$(S_{cc}^{-1}+S_n^{-1})U=S_{cc}^{-1}U_{cc}+S_n^{-1}\overline{U} \tag{2.30}$$

The iterative algorithm is then used as following

$$U^{k+1}=[S_{cc}^{-1}+S_n^{-1}]^{-1}[S_{cc}^{-1}U_{cc}+S_n^{-1}\overline{U}^k] \tag{2.31}$$

with the starting iteration value of U be U^0, equal to the conservation estimate $U^0=U_{cc}$.

2.1.3 Correlation-feedback Approach

The third approach studied in this thesis is the newly developed method by Pan et al. [3]. This approach is based on the correlation-based approach and the idea of feedback, that is why it is called correlation-feedback approach.

Clearly, the procedure of this approach can be viewed as two steps, one is correlation step, and the other is propagation step.

Unlike correlation-based approach, in correlation step, a new continuous two-dimension function $f(i,j)$ is defined as an extension of the real digital image $I(i,j)$.

As for $image$ 1, we define it as following

$$f_1(i_1,j_1)=I_1(i_1,j_1) \tag{2.32}$$

where i_1 and j_1 are indexes of I_1. Then, for $image2$, using bilinear interpolation, we define

$$\begin{aligned}&f_2(i_2-u_n(i_2,j_2),j_2-v_n(i_2,j_2))\\&=(1-a)[(1-b)I_1(int(x),int(y))+bI_1(int(x),int(y)+1)]\\&+a[(1-b)I_1(int(x)+1,int(y))+bI_1(int(x)+1,int(y)+1)]\end{aligned} \tag{2.33}$$

where $int(x)=int(i_2-u_n)$; $int(y)=int(j_2-v_n)$; $a=i_2-u_n-int(x)$; $b=j_2-v_n-int(y)$. Information about $\overline{U}$ is known from the difference between the expected value of $(I_2(i_2, j_2))$ and $I_2(i_2, j_2)$. A correlation-window W_p of size $(2n+1)*(2n+1)$ is formed around the pixel at location (x, y) in the first image I_1, and a search-window W_s of size $(2N+1)*(2N+1)$ is formed around the pixel at the location in the second image I_2. The $(2N+1)^2$ samples of error-distribution are computed by using the sum-of-squared-differences as

$$\varepsilon(u,v) = \sum_{x=-n}^{n} \sum_{y=-n}^{n} (I_2(i_2+x, j_2+y) - f_2(i_2+x-u, j_2+y-v))^2 \qquad (2.34)$$

The $(2N+1)^2$ samples of response-distribution are computed as follows

$$R_c(u,v) = e^{-k\varepsilon(u,v)} \qquad (2.35)$$

Based on the weighted-least-squares estimation

$$u_c(i_2, j_2) = \frac{\sum_u \sum_v R_c(u,v)u}{\sum_u \sum_v R_c(u,v)} \qquad (2.36)$$

$$v_c(i_2, j_2) = \frac{\sum_u \sum_v R_c(u,v)v}{\sum_u \sum_v R_c(u,v)} \qquad (2.37)$$

In the propagation step, forming the neighborhood window of size $(2w+1)*(2w+1)$, mapping the velocities of the these $(2w+1)*(2w+1)$ pixels to the points (u_i, v_i) in the u-v space, choosing the weight $w(x, y)$ as a Gaussian mask, we get

$$u_{n+1} = \sum_{x=-N}^{N} \sum_{y=-N}^{N} w(x,y)u_c(i_2+x, j_2+y) \qquad (2.38)$$

$$v_{n+1} = \sum_{x=-N}^{N} \sum_{y=-N}^{N} w(x,y)v_c(i_2+x, j_2+y) \qquad (2.39)$$

Figure 2.1.3 shows a block diagram of this framework.

This iterative procedure should be terminated when $|u_{n+1} - u_n|$ and $|v_{n+1} - v_n|$ are greater than a threshold, expected value of $(I_2(i,j))$ is updated in the observe stage according to $u_{n+1}(i,j)$, $v_{n+1}(i,j)$ and $I_1(i,j)$. Since the algorithm is convergent (see [3] for the hairy detail), (u_n, v_n) tends to the true optical-flow vector.

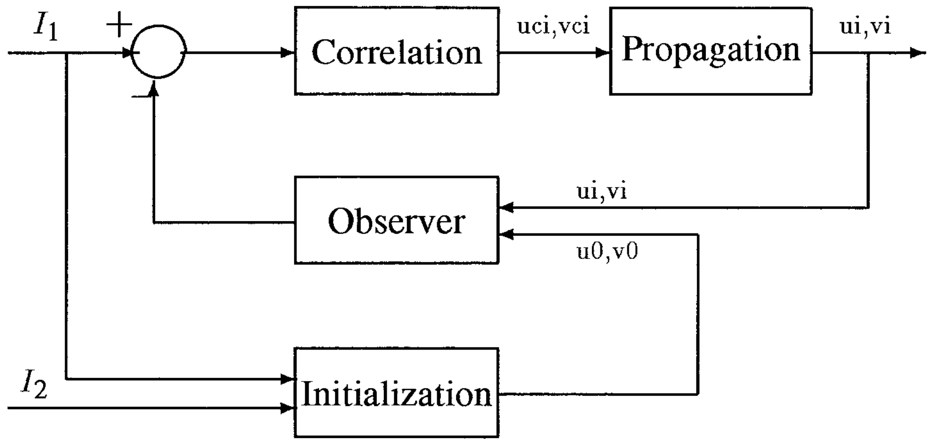


Figure 2.1 Schematic Diagram for Correlation-Feedback Approach

2.2 Experimental Techniques

Experiments have been conducted to examine the performance of these three different techniques on real image sequences. There are three different image sequences which are taken in different settings and used to conduct three different experiments. Before discussing the results, the settings and the image sequences as well as error measurements used in experiments are described.

The image sequences used here are taken by a CCD TV camera and through an peripheral device attached on a Sun SPARCstation made by DATACUBE Inc. The size of images is 64×64 $(pixel \times pixel/frame)$. The detail procedure of how the 64×64 $(pixel \times pixel/frame)$ image sequence is created is illustrated as follows: 1) By using the CCD TV camera and DATACUBE electronic circuit boards, an image of 512×512 $(pixel \times pixel/frame)$ is created. 2) The 512×512 $(pixel \times pixel/frame)$ image is too big for computing , so it is cut into a 256×256 $(pixel \times pixel/frame)$ image 3) In order to decrease the influence of noise, the 256×256 $(pixel \times pixel/frame)$ image is compressed into a 64×64 $(pixel \times pixel/frame)$ image with sub-sampling method, that is, first, a 256×256 $pixel \times pixel/frame$ image is divided into 64×64 blocks, with each block having $4 \times 4 pixels$, i.e., one block corresponding to a pixel in

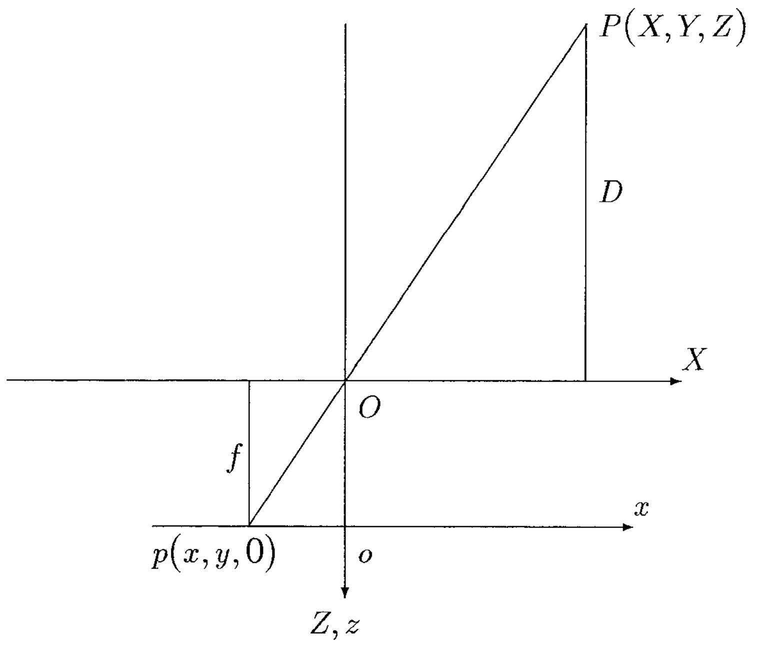


Figure 2.2 The Plane of Perspective Projection

the 64 × 64 $(pixel \times pixel/frame)$ image, then, assigning the average intensity value of the 4 × 4 $(pixel \times pixel/frame)$ block to the intensity value of the corresponding pixel in the 64 × 64 $(pixel \times pixel/frame)$ image.

In order to use error measurement to evaluate the performance of the proposed approaches, not only the real optical flow but also the correct optical flow is needed. The perspective projection between an object in 3-d world space and an image in 2-d image plane is $\frac{x}{f} = \frac{X}{D}$. All these parameters are illustrated in the perspective projection plane (see Figure 2.2), where X is the distance of one point moves in the object, D is the distance between the optical center of the camera and the object, x is the distance of the pixel corresponding to the same point in that object moves in the image plane , f is the focal length of camera. According to this perspective projection, the correct optical field can be calculated.

2.2.1 Experiments I, II and III

- Experiment I

Experiment I described here uses a sphere standing before a white post which is used as background. The sphere is remained still while the camera and white post are clockwise rotated at the same speed, which means that the white post keeps still relative to the camera. Refer to the perspective projection plane, the clockwise rotation velocity is $\theta = -2.5^o/frame$ (assuming the rotation should be counterclockwise), the focal length of the CCD camera is $f = 12.5mm$, the distance between the center of the sphere and the optical center of the camera is $D = -1080mm$. According to the perspective projection transformation, the true optical flow $\vec{U}_c(u_c, v_c)$ can be calculated. This algorithm can be illustrated as below:

Given one point $p(x, y)$ in the image plane, the corresponding point $P(X, Y, Z)$ in 3-d world space can be calculated with using the perspective projection. When the image sequence moves from one frame to the next frame, the point $P(X, Y, Z)$ moves to the point $P_2(X_2, Y_2, Z_2)$. The relation between $P(X, Y, Z)$ and $P_2(X_2, Y_2, Z_2)$ is fixed when the experimental setting is given. So, the corresponding point $p_2(x_2, y_2)$ in the image plane can be calculated with using the perspective projection again. Assuming the time interval between one frame and the next frame is one second, the velocity can be illustrated as $u_c = \Delta x = x_2 - x$ $(mm/second)$ and $v_c = \Delta y = y_2 - y$ $(mm/second)$. Practically, the velocity is expressed with unit $(pixel/second)$ as following

$$u_c = \frac{\Delta x}{l_x} = \frac{x_2 - x}{l_x}(pixel/second)$$

$$v_c = \frac{\Delta y}{l_y} = \frac{y_2 - y}{l_y}(pixel/second)$$

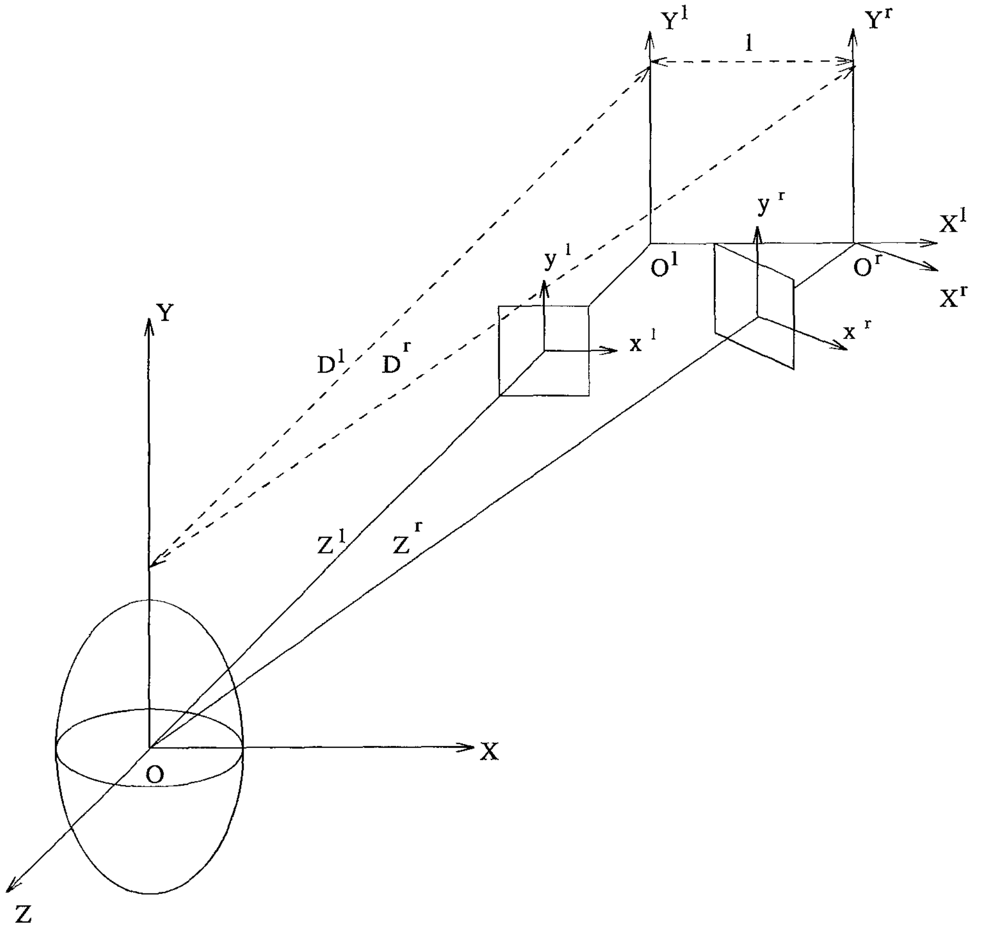
Refer to the Appendix for the final correct optical flow result. The velocity along x-direction ranges from 0.0 $(pixel)$ to -0.9444 $(pixel)$. The velocity along y-direction ranges from -0.04583 $(pixel)$ 0.04583 $(pixel)$. It's worth to notice that the unit along x direction $(l_x = 0.05588(mm/pixel))$ and the unit along y direction $(l_y = 0.04657(mm/pixel))$ in the image plane are not the same by calibration, the projection of sphere into the image plane is actually an ellipse instead of a circle.

Figure 2.3 shows the setting for this experiment. Figure 2.4-- 2.7 is an image sequence which is generated in this setting.

- Experiment II

Experiment II described here uses a texture post standing above a table. The CCD camera moves normal to its line of sight, the moving line is parallel to the plane of the texture post. In the perspective projection plane, the CCD camera's moving distance is $l = -2.2(mm/frame)$, the focal length of the camera is $f = 30mm$, the distance between the texture post plane and the optical center of the camera is $D = -850mm$. By calibration, it is known that the unit along x direction is $l_x = 0.05588(mm/pixel)$. According to the perspective projection $\frac{f}{x} = \frac{D}{X}$, and the algorithm for calculating correct optical flow (u_c, v_c) which is illustrated in Experiment I, the correct optical flow can be calculated as below. Also assuming the time interval between one frame and the next frame is one second.

$$\begin{aligned} \frac{x}{f} &= \frac{X}{D} & \frac{y}{f} = \frac{Y}{D} \\ X &= \frac{x D_1}{f} & Y = \frac{y D_1}{f} \\ X_2 &= X - l & Y_2 = Y \\ D_2 &= D_1 = D \end{aligned}$$



$o^l - x^l - y^l, o^r - x^r - y^r$	-- image planes
$O - X - Y - Z$	-- 3-d coordinate system
$O^l - X^l - Y^l - Z^l$	-- 3-d auxiliary coordinate system
$O^r - X^r - Y^r - Z^r$	-- 3-d auxiliary coordinate system

Figure 2.3 Setting in Experiment I (sphere)

Figure 2.4 The First Image Used in Experiment I (sphere)

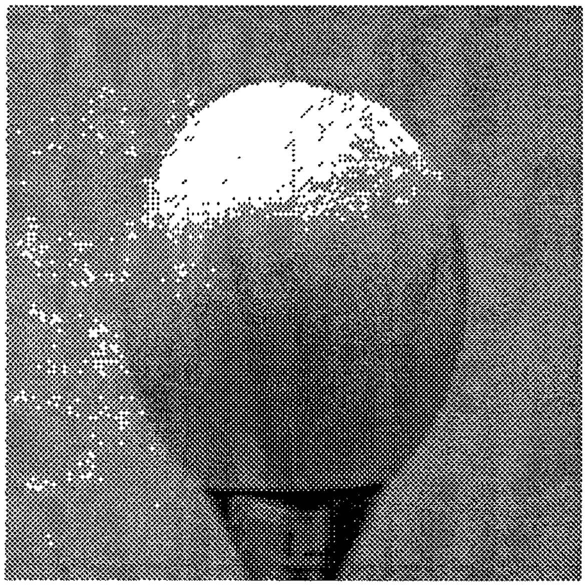


Figure 2.5 The Second Image Used in Experiment I (sphere)

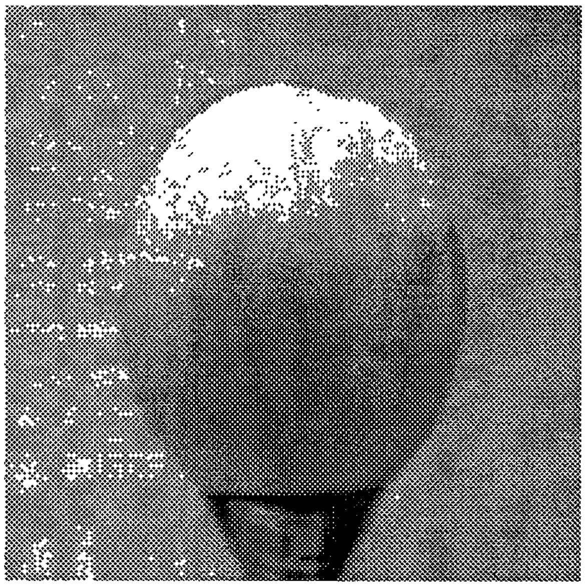


Figure 2.6 The Third Image Used in Experiment I (sphere)

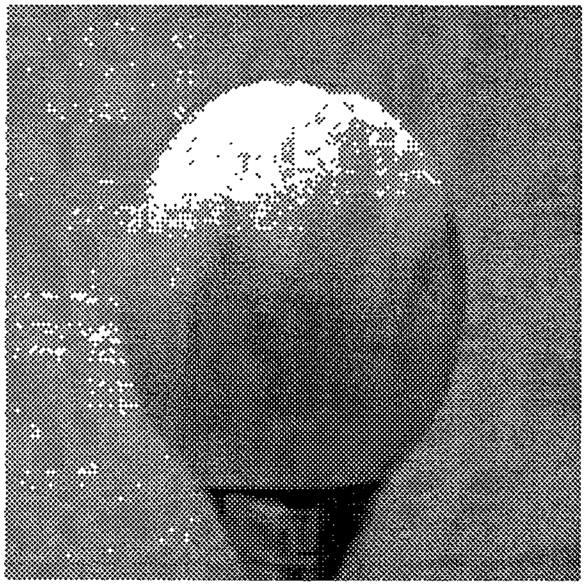


Figure 2.7 The Fourth Image Used in Experiment I (sphere)

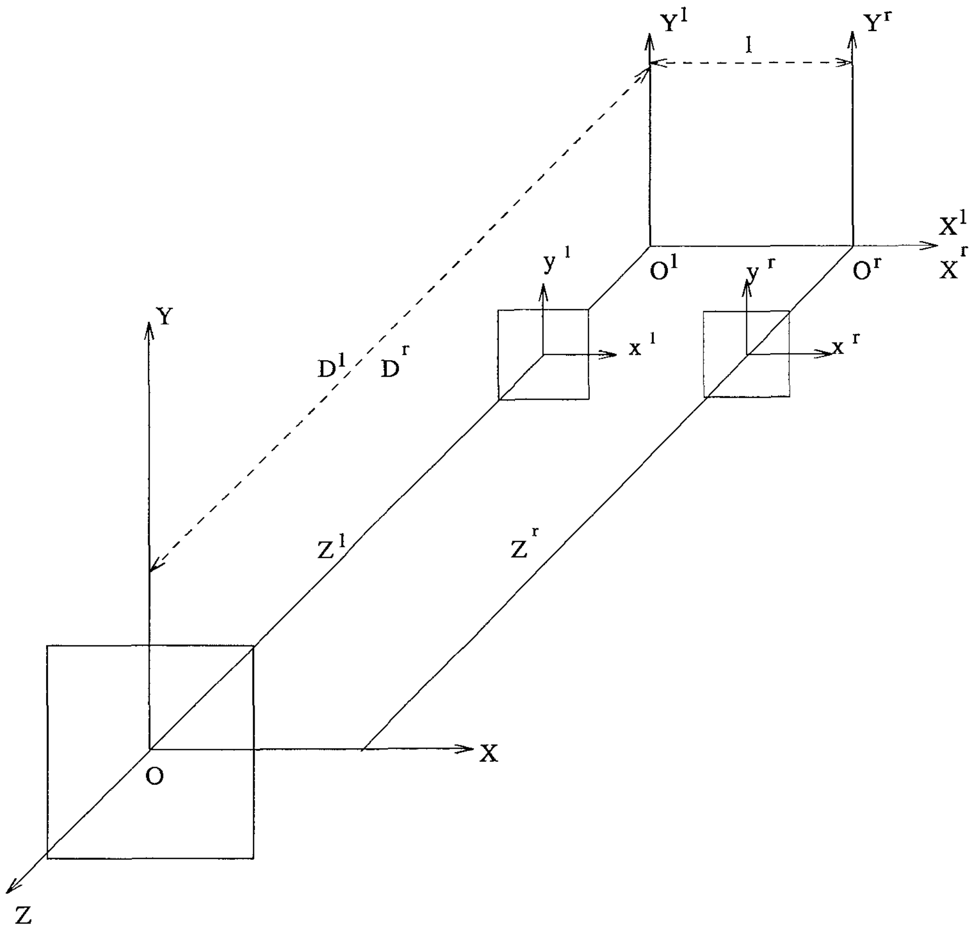
$$\begin{aligned}
x_2 &= \frac{X_2 f}{D_2} = \frac{(X-l)\,f}{D} \\
y_2 &= \frac{Y_2 f}{D_2} = \frac{Y f}{D} \\
\Delta x &= x_2 - x = \frac{(X-l)\,f}{D} - \frac{X f}{D} = -\frac{l f}{D} \\
\Delta y &= y_2 - y = \frac{Y f}{D} - \frac{Y f}{D} = 0
\end{aligned}$$

$$\begin{aligned}
u_c &= \frac{\Delta x}{l_x} = -\frac{l f}{D\, l_x} = -1.3895\,(pixel/second) \\
v_c &= \frac{\Delta y}{l_y} = 0\,(pixel/second)
\end{aligned}$$

Figure 2.8 shows setting for this experiment. Figure 2.9-- 2.12 is an image sequence that is obtained corresponding to this setting.

- Experiment III

Experiment III described here uses a texture standing above a table, similarly to the Experiment II. The CCD camera moves normal to its line of sight, but the moving line is not parallel to the texture post plane, the texture post is on a slant to the camera. The degree between the slant and the normal of the sight line is $\theta = 35^o$. In the perspective projection plane, the camera's moving distance is $l = -2.2\ (mm/frame)$, the focal length of the camera is $f = 30mm$, the distance between the center of the slant and the optical center of the camera is $D = -850mm$. By calibration,it is known that the unit along x direction is $l_x = 0.05588\ (mm/pixel)$ and the unit along y direction is $l_y = 0.0465664\ (mm/pixel)$. According to the perspective projection which is $\frac{f}{x} = \frac{D^*}{X}$ while D^* changes with different point in the slant, and the algorithm for calculating correct optical flow (u_c, v_c) which is illustrated in Experiment I, the correct optical flow can be calculated as below. Also assuming the time interval between one



$o^l - x^l - y^l, o^r - x^r - y^r$	-- image planes
$O - X - Y - Z$	-- 3-d coordinate system
$O^l - X^l - Y^l - Z^l$	-- 3-d auxiliary coordinate system
$O^r - X^r - Y^r - Z^r$	-- 3-d auxiliary coordinate system

Figure 2.8 Setting in Experiment II (plane)

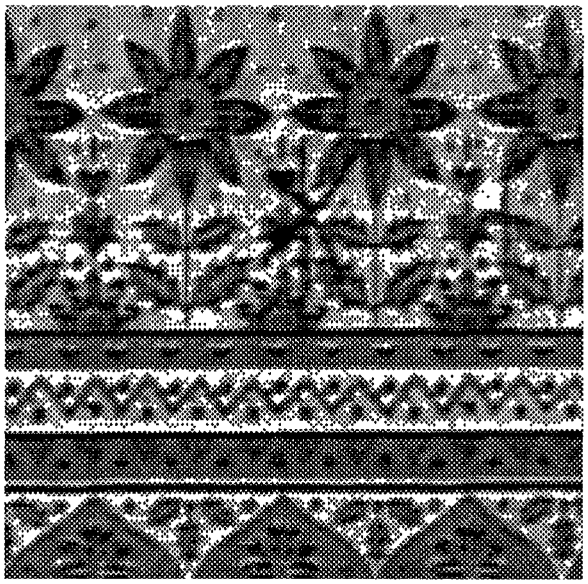


Figure 2.9 The First Image Used in Experiment II (plane)

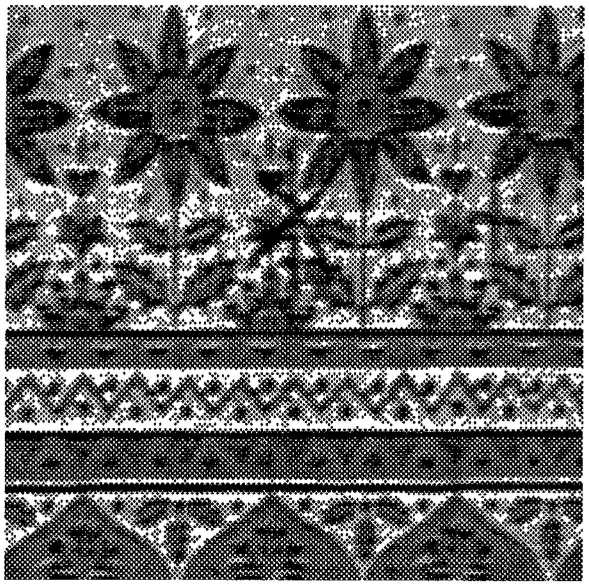


Figure 2.10 The Second Image Used in Experiment II (plane)

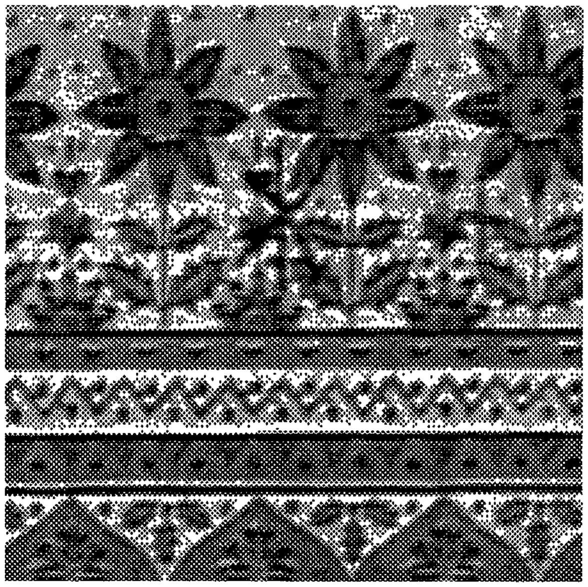


Figure 2.11 The Third Image Used in Experiment II (plane)

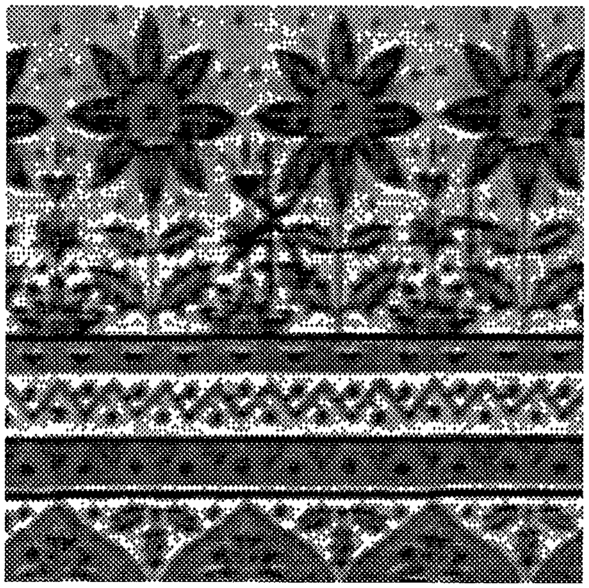


Figure 2.12 The Fourth Image Used in Experiment II (plane)

frame and the next frame is one second.

$$X = \frac{xD_1}{f} \qquad Y = \frac{yD_1}{f}$$

$$where\ D_1 = D + X tan\theta = D + \frac{xD_1}{f} tan\theta$$

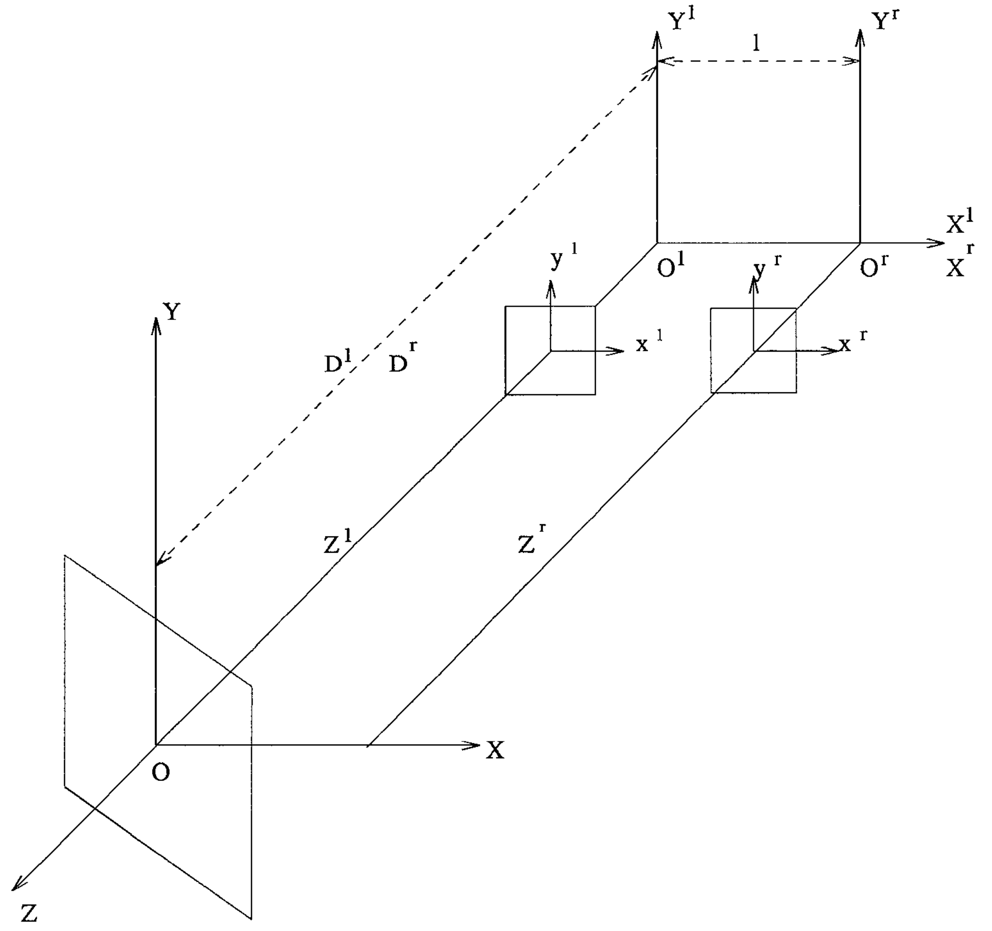
$$\begin{aligned}
D_1 &= \frac{D}{1 - \frac{x\,tan\theta}{f}} \\
X_2 &= X - l \qquad Y_2 = Y \\
D_2 &= D_1 - l\,tan\theta \\
x_2 &= \frac{X_2 f}{D_2} \qquad y_2 = \frac{Y_2 f}{D_2}
\end{aligned}$$

$$\begin{aligned}
\Delta x &= x_2 - x = \frac{(X-l)f}{D_1 - l\,tan\,\theta} - \frac{X_1 f}{D_1} \\
&= \frac{-l\,f\,D_1 + X\,f\,l\,tan\,\theta}{D_1(D_1 - l\,tan\,\theta)} \\
&= \frac{-f\,l(1 - \frac{x}{f}tan\theta)}{\frac{D}{1-\frac{x\,tan\,\theta}{f}} - l\,tan\,\theta}
\end{aligned}$$

$$\begin{aligned}
\Delta y &= y_2 - y = \frac{Yf}{D_2} - \frac{Yf}{D_1} \\
&= Y f(\frac{1}{D_1 - l\,tan\,\theta} - \frac{1}{D_1}) \\
&= \frac{Y\,f\,l\,tan\,\theta}{D_1(\,D_1 - l\,dtan\,\theta\,)} \\
&= \frac{y\,l\,tan\,\theta}{\frac{D}{1-\frac{x\,tan\,\theta}{f}} - l\,tan\,\theta}
\end{aligned}$$

$$\begin{aligned}
u_c &= \frac{\Delta x}{l_x}\,(pixel/second) \\
v_c &= \frac{\Delta y}{l_y}\,(pixel/second)
\end{aligned}$$

From the above equations, the correct optical flow along x-direction ranges from -1.2847 to -1.5138 $(pixel/second)$, the correct optical



$o^l - x^l - y^l, o^r - x^r - y^r$	-- image planes
$O - X - Y - Z$	-- 3-d coordinate system
$O^l - X^l - Y^l - Z^l$	-- 3-d auxiliary coordinate system
$O^r - X^r - Y^r - Z^r$	-- 3-d auxiliary coordinate system

Figure 2.13 Setting in Experiment III (slant)

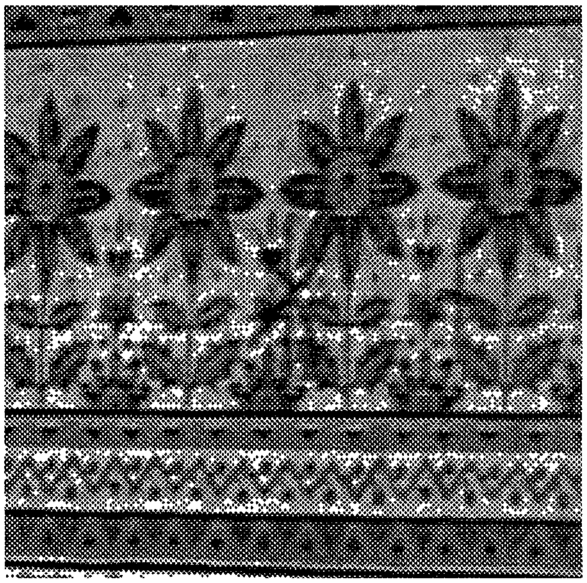


Figure 2.14 The First Image Used in Experiment III (slant)

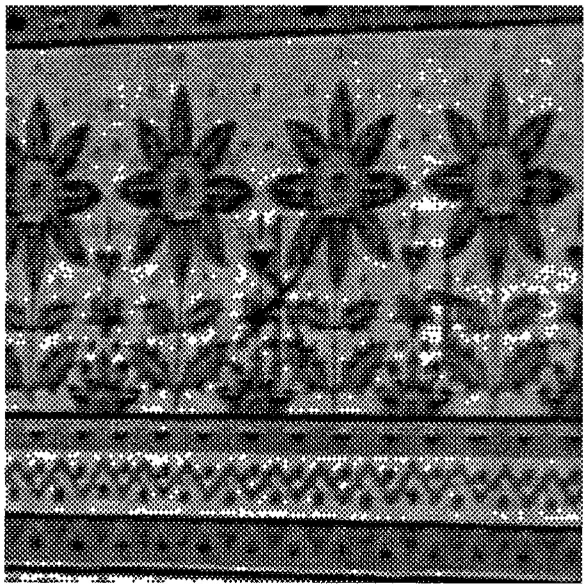


Figure 2.15 The Second Image Used in Experiment III (slant)

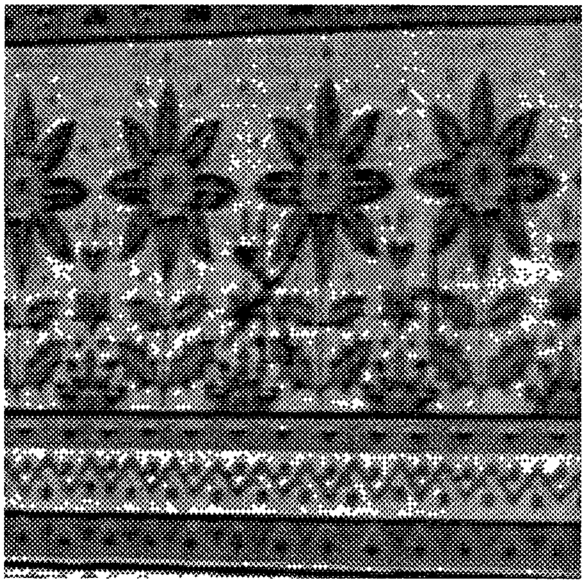


Figure 2.16 The Third Image Used in Experiment III (slant)

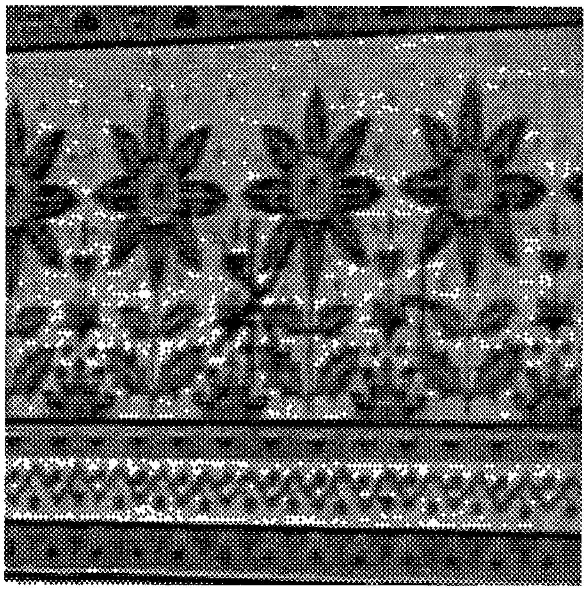


Figure 2.17 The Fourth Image Used in Experiment III (slant)

flow along y-direction ranges from –0.0540 to 0.0605 $(pixel/second)$. Figure 2.13 shows setting for this experiment. Figure 2.14-- 2.17 is an image sequence that is created in this setting.

2.2.2 Error Measurements

Two error measurements are used here to analyze the performance of these three approaches for image flow determination. One measurement is the relative error measurement, and the other is angular error measurement. They are described respectively in the following.

- Relative error ε_e.

  Let $\vec{U_c}(u_c, v_c)$ denote correct optical flow vector, $\vec{U_e}(u_e, v_e)$ denote the estimate optical flow vector, the relative error between the correct velocity $\vec{v_c}$ and the estimate $\vec{v_e}$ is

$$\varepsilon_e = \frac{\sqrt{\sum_{i=0}^{n}\sum_{j=0}^{n}\left((u_c(i,j) - u_e(i,j))^2 + (v_c(i,j) - v_e(i,j))^2\right)}}{\sqrt{\sum_{i=0}^{n}\sum_{j=0}^{n}(u_c(i,j)^2 + v_c(i,j)^2)}}$$

- Angular measure of error ψ_e [4].

  Assuming that velocity can be written as displacement per unit time as in v $= (u,v)^T pixels/frame$, or as a spatiotemporal direction vector $(u,v,1)^T$ in units of (pixel,pixel,frame). Velocity is obtained from the direction vector by dividing through by the third component. When velocity is viewed (and measured) as orientation in space-time domain, it is natural to measure errors as angular deviations from the correct space-time orientation, we call it angular measure of error. Therefore, let velocities v $= (u,v)^T$ be represented as 3-d unit direction vector, $\vec{\mathrm{v}} = \frac{1}{\sqrt{u^2+v^2+1}}(u,v,1)^T$. The error between the correct velocity $\vec{v_c}$ and an estimate $\vec{v_e}$ is

$$\psi_e = arccos(\vec{v_c} \cdot \vec{v_e})$$

This angular measure of error is convenient in some cases because it can handle very large and very small speeds without the amplification inherent in a relative measure of vector differences. However, it does have some bias. For example, directional errors at small speeds do not give as large an angular error as similar directional errors at higher speeds. A complementary measure is also available for errors in measurements of normal (component) velocity [4].

2.3 Experimental Results

By using the relative error and angular error measurement, the performances of the three different approaches are analyzed quantitatively. The results are obtained by using real image sequences as input image sequences. Three different computer programs which implement these three different approaches are used here in these experiments. The program for Singh's approach is provided by J.L.Barron (University of Western Ontario, Canada), the programs for Horn and Schunck's approach and Pan's approach are provided by J.N.Pan (New Jersey Institute of Technology, U.S.A). I've studied these programs intensively and extensively, and fixed some bugs in these programs, I also enhanced them and made some modification to make them suitable of my experiments. As the requirement of these programs, the input images should be SUN raster files for Singh's program, and input images should be float-data files for Horn and Schunck's, and Pan's programs. Also, Horn and Schunck's program just need two images as the input image sequence, and Singh's and Pan's programs need three images as the input image sequence.

Since the results for different programs depend on how the parameters are chosen, it is very important to select the appropriate parameters for different approaches before running the programs. As to the Singh's algorithm, the parameters N, k, n and w have to be established in order to compute response-distribution (refer to the section for Singh's algorithm as a represent for correlation-based approach for more detail). The choice of N --- the search-window size --- depends on the maximum possible displacement of a pixel between two frames. If the displacement is small, $N = 2$ (i.e., a 5×5 search window) is appropriate. If the displacement is large, one can still use $N = 2$ with a hierarchical search strategy. The parameter k --- parameter for the exponential function for converting error-distribution into response-distribution --- is essentially a normalization factor, it is chosen so to make the maximum response-distribution a fixed number which usually close to unity. The values of n --- the correlation-window size --- and w --- the neighborhood size --- are decided on the basis of how many neighbors should contribute to the estimation of velocity of the point under consideration. Too small neighborhood leads to noisy estimates, and too large neighborhood leads to distortion caused by excessively smoothing. Empirically, $n = 1$ and $w = 1$ (i.e., a 3×3 window) appears appreciate. Also, when solving the linear equations for the result of u,v, inverse of various matrices becomes hard when one or more of their eigenvalue are zero or very small. Singular-value-decomposition is used for matrix-inversion in order to solve this kind problem. As to Horn and Schunck's algorithm, the parameter α --- a weighting factor between the measurement for the rate of change of image brightness and the measurement for the departure from smoothness in the velocity flow --- should be roughly equal to the expected value of noise in the estimate of $I_x^2 + I_y^2$. (Consulting the section for Horn and Schunck's algorithm as a represent for gradient-based approach for more detail). From the equations shown in that section, it is well known that α^2 plays a significant role only for areas where the brightness gradient is small, preventing hazardous adjustments to the estimated image flow velocity caused by noise in the estimated derivatives. As to Pan's algorithm, the parameters α, N, k, n and w should be chosen by combining the criteria of Singh and Horn and Schunck's. For all these three programs, criteria have been established to stop the iterative update processes.

The input real images are 64×64 $(pixel \times pixel)/frame$. The central 40×40 $pixel \times pixel$ optical flow vector arrays are used to compute the error used in the program provided by J.L.Barron, so for the consistency, all the input for error analysis of all these approaches are the central 40×40 $pixel \times pixel$ optical flow vector. Two results are obtained for each experiments by two different input image sequences.One result is shown in Table 2.1 and Table 2.2 ,which is obtained by using the first two or three images and corresponds to the velocity u^l and v^l, and the other result is shown in Table 2.3 and Table 2.4, which is obtained by using the second two or three images and corresponds to the velocity u^r and v^r. The notations u^l, v^l, u^r and v^r used here will be explained in next chapter.

Following is some notations used in these Tables. α --- a weight factor between the measurement for the rate of change of image brightness and the measurement for the departure from smoothness in the velocity flow. sw --- means the search window. cw --- means the correlation window. gm --- means the Gaussian mask. deviat --- means the deviation for the statistical characteristic. 1,2,3,4 --- mean the 1st, 2nd, 3rd and 4th image used in these experiments. * --- means the best approach among these three approaches corresponding to one specific experiment setting.

From the result shown in the Table 2.1, Table 2.2, Table 2.3 and Table 2.4, the following observation can be achieved.

Table 2.1 Results of Relative Error (as to u^l,v^l)

	Gradient-based approach Using image 2,3	Correlation-based approach Using image 1,2,3	Correlation-feedback approach Using image 1,2,3

	Iteration = 128
	$\alpha = 10$
		Iteration = 25
		sw = 9×9
		cw = 5×5
		gm = 5×5
			Iteration = 12
			Horn-iteration = 30
			sw = 5×5
			cw = 3×3
			gm = 3×3

	ε_e	ε_e	ε_e
Sphere	0.5466 *	0.7693	1.647
Plane	0.2611	0.2716	0.1691 *
Slant	0.3669	0.5389	0.2188 *

Table 2.2 Results of Angular Error (as to u^l,v^l)

	Gradient-based approach Using image 2,3		Correlation-based approach Using image 1,2,3		Correlation-feedback approach Using image 1,2,3	
	Iteration = 128 $\alpha = 10$		Iteration = 25 sw = 9×9 cw = 5×5 gm = 5×5		Iteration = 12 Horn-iteration = 30 sw = 5×5 cw = 3×3 gm = 3×3	
	ψ_e		ψ_e		ψ_e	
	average	deviat.	average	deviat.	average	deviat.
Sphere	14.699	7.8545	10.125	11.611	32.411	15.251
Plane	8.2221	7.1052	8.9019	0.6881	3.945	2.2408
Slant	12.507	10.081	15.094	18.408	6.4788	6.4275

Table 2.3 Results of Relative Error (as to u^r,v^r)

	Gradient-based approach Using image 3,4	Correlation-based approach Using image 2,3,4	Correlation-feedback approach Using image 2,3,4

	Iteration = 128
	$\alpha = 10$
		Iteration = 25
		sw = 9×9
		cw = 5×5
		gm = 5×5
			Iteration = 12
			Horn-iteration = 30
			sw = 5×5
			cw = 3×3
			gm = 3×3

	ε_e	ε_e	ε_e
Sphere	0.5242 *	0.6873	1.2073
Plane	0.5111	0.6505	0.1456 *
Slant	0.4687	0.6066	0.2208 *

Table 2.4 Results of Angular Error (as to u^r,v^r)

	Gradient-based approach Using image 3,4		Correlation-based approach Using image 2,3,4		Correlation-feedback approach Using image 2,3,4	
	Iteration = 128 $\alpha = 10$		Iteration = 25 $= 9 \times 9$ cw $= 5 \times 5$ gm $= 5 \times 5$		Iteration = 12 Horn-iteration = 30 sw $= 5 \times 5$ cw $= 3 \times 3$ gm $= 3 \times 3$	
	ψ_e		ψ_e		ψ_e	
	average	deviat.	average	deviat.	average	deviat.
Sphere	13.755	8.3677	7.5548	9.7462	19.833	16.676
Plane	21.852	9.6763	21.731	22.612	4.0385	2.9386
Slant	19.419	9.9719	17.093	21.411	6.4048	5.3374

- The gradient-based approach is better than the correlation-based and correlation-feedback approaches when the boundary is not too sharp as shown in Experiment I --- Sphere experiment.

- The correlation-feedback approach is appreciate to texture post which has discontinuities boundary as shown in Experiment II and III --- Plane and Slant experiments.

- The correlation-based approach has an inherent shortcoming which is lacking of the ability to estimate not integer number of displacement. In Experiment I, the correct optical flow is almost integer, and in Experiment II and III, the correct optical flow are not integer numbers.

Also, vector diagrams for flow field are developed to give more visual comparison among these different approaches. It's worth noticing that only the central 40×40 $(pixel \times pixel)$ optical flow vector arrays are created in the program provided by J.L.Barron. For the consistency, all the input for error analysis of these approaches are the central 40×40 $(pixel \times pixel)$ optical flow vector, that's also the input for the creation of vector diagrams for flow field which are shown from Figure 2.18 to Figure 2.29. But for the Sphere

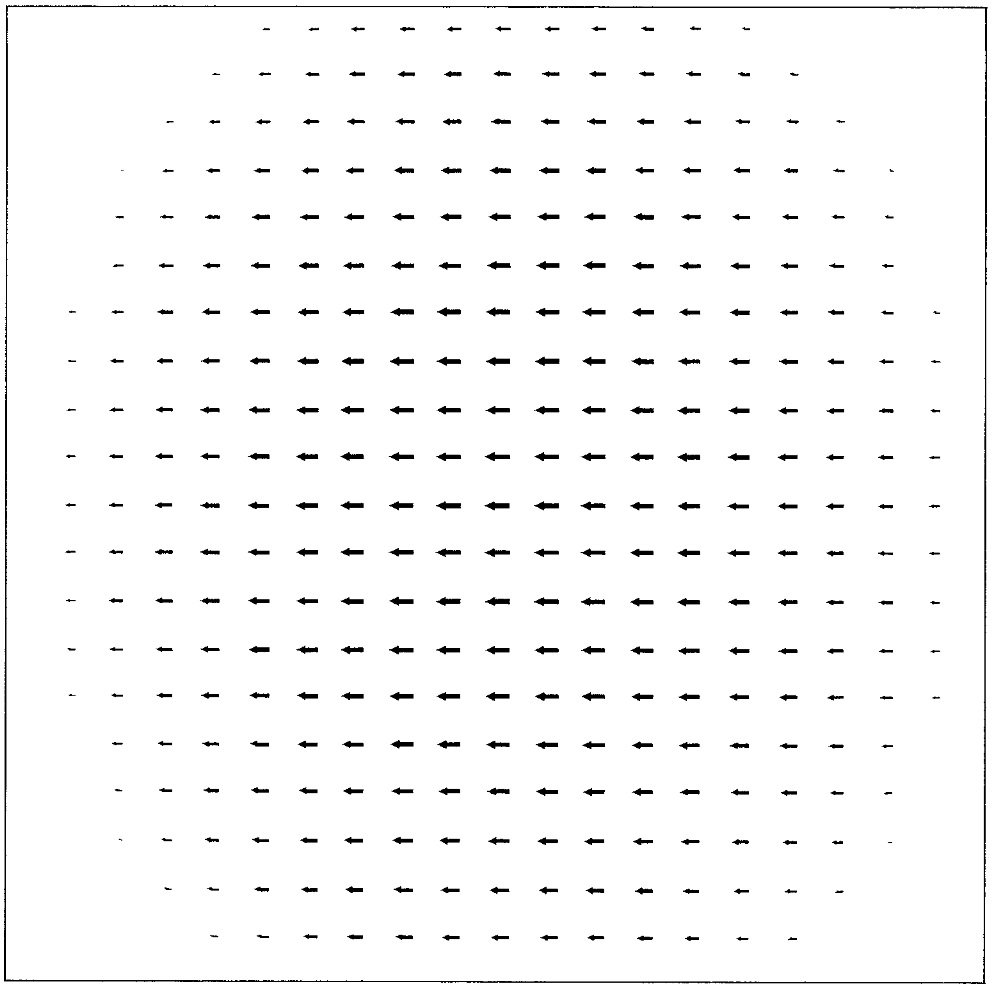


Figure 2.18 Correct flow field in Sphere Experiment

Experiment, the tendency of flow field is not obvious if the offset size is ignored. So in Figure 2.30, Figure 2.31 and Figure 2.32, the inputs of vector diagrams are 64×64 $(pixel \times pixel)$ optical flow vector.

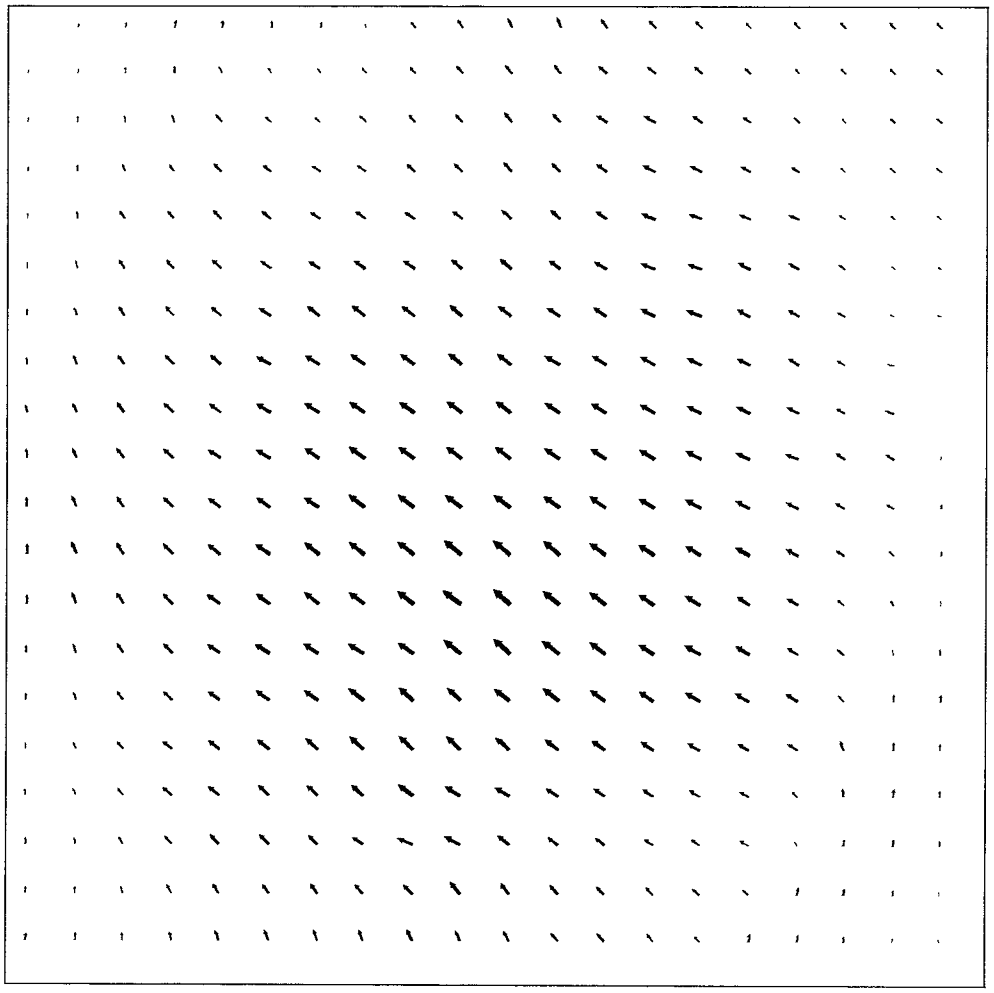


Figure 2.19 Calculated flow field in Sphere Experiment (Horn)

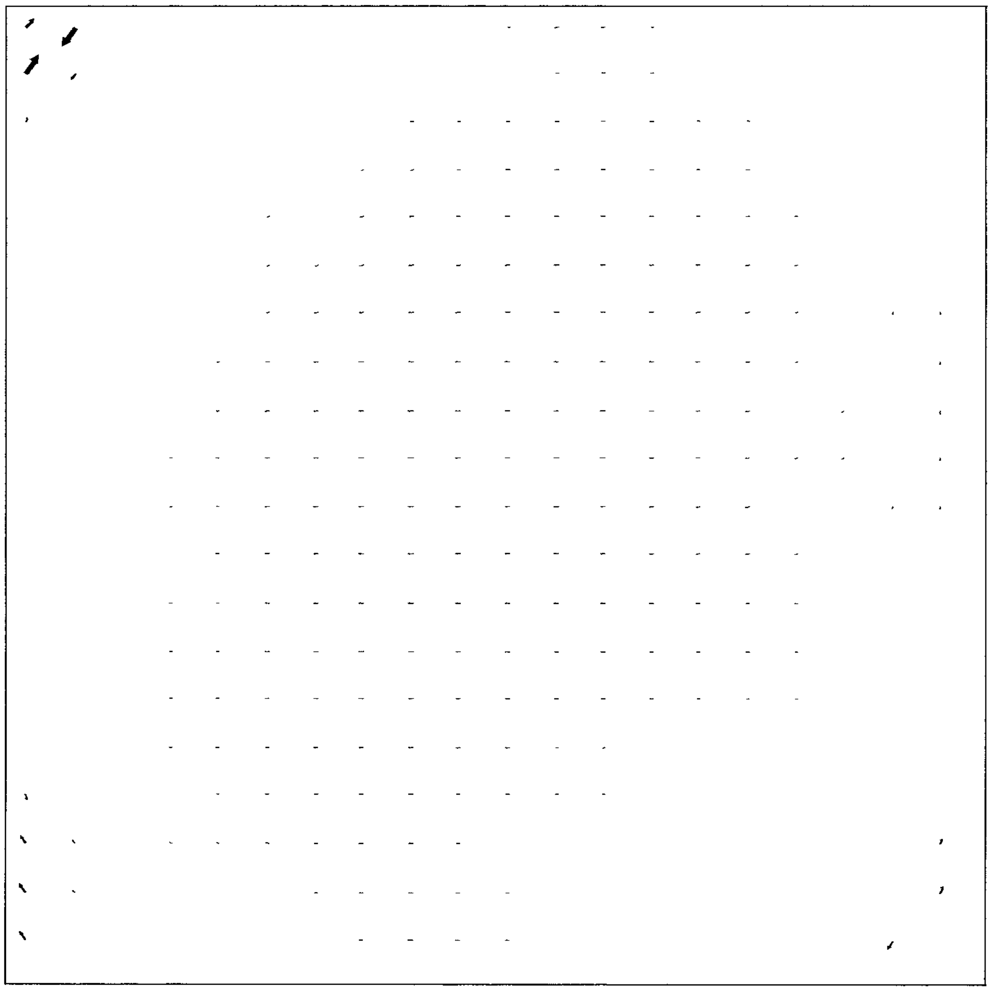


Figure 2.20 Calculated flow field in Sphere Experiment (Sıngh)

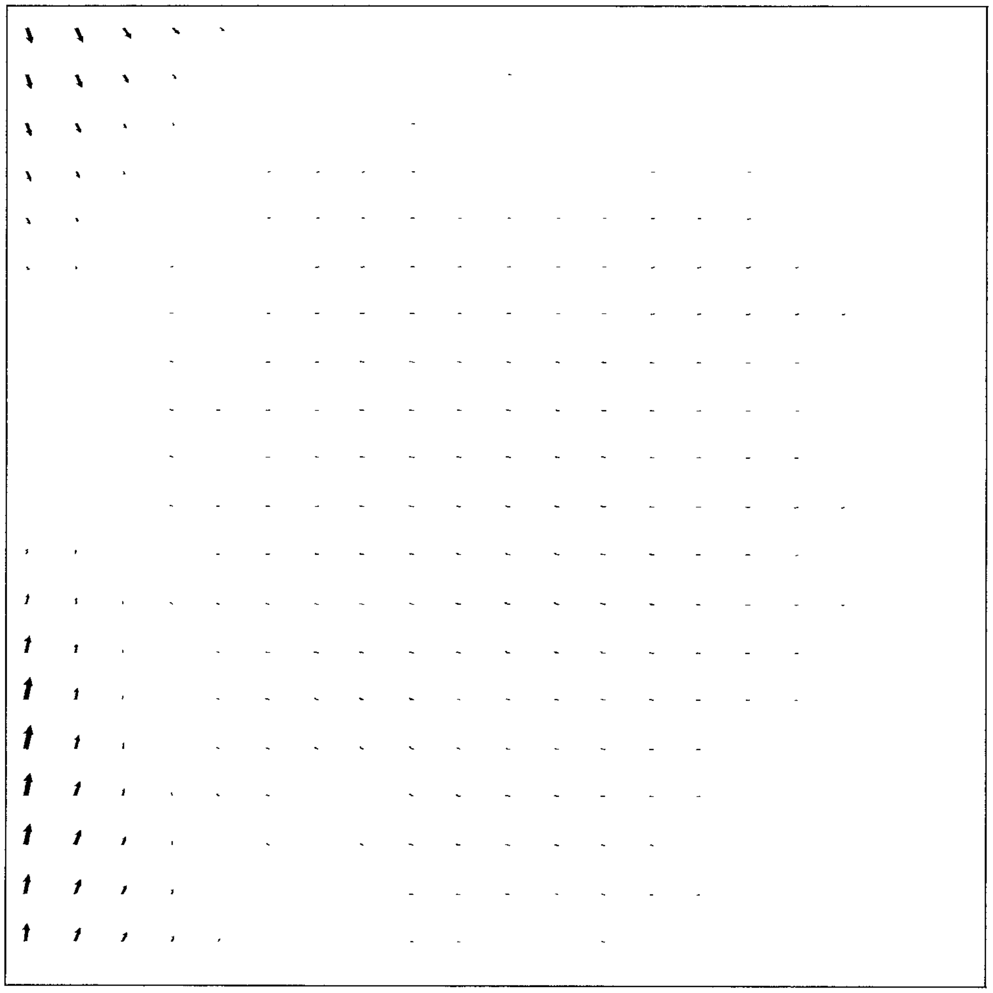


Figure 2.21 Calculated flow field in Sphere Experiment (Pan)

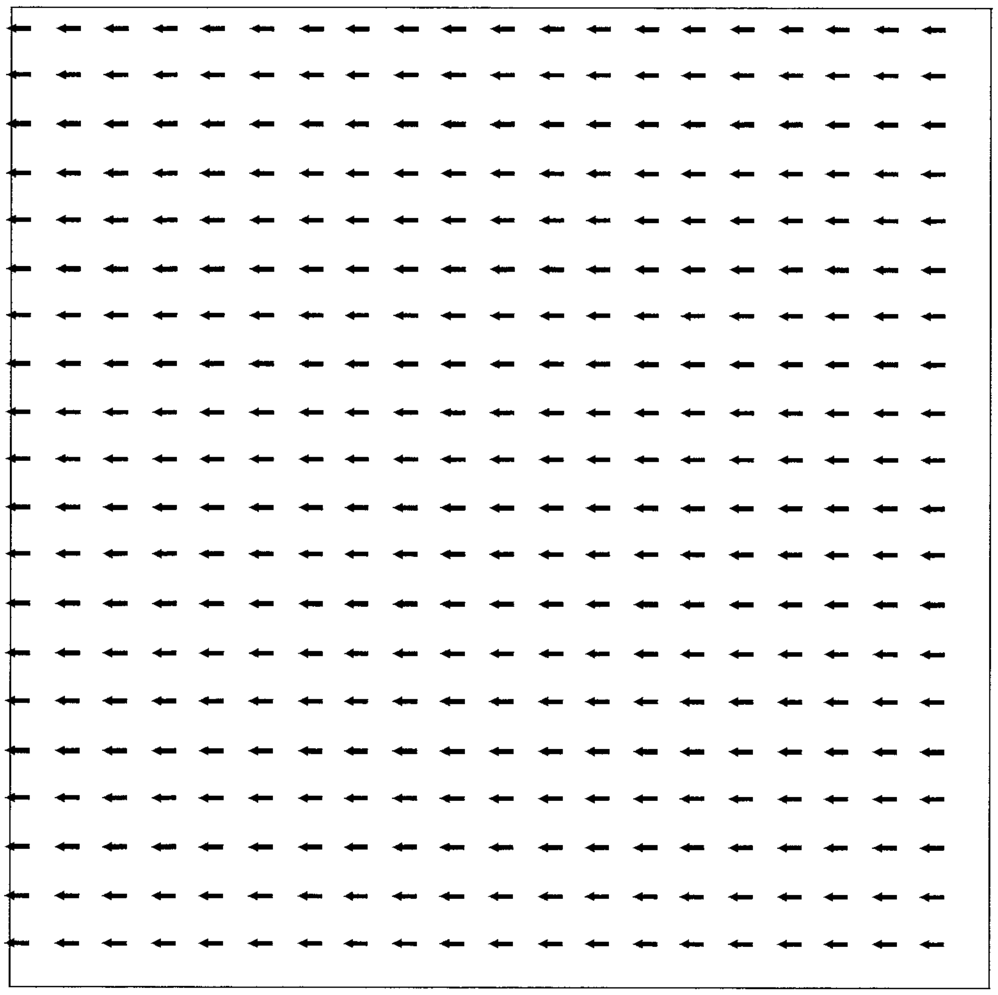


Figure 2.22 Correct flow field in Plane Experiment

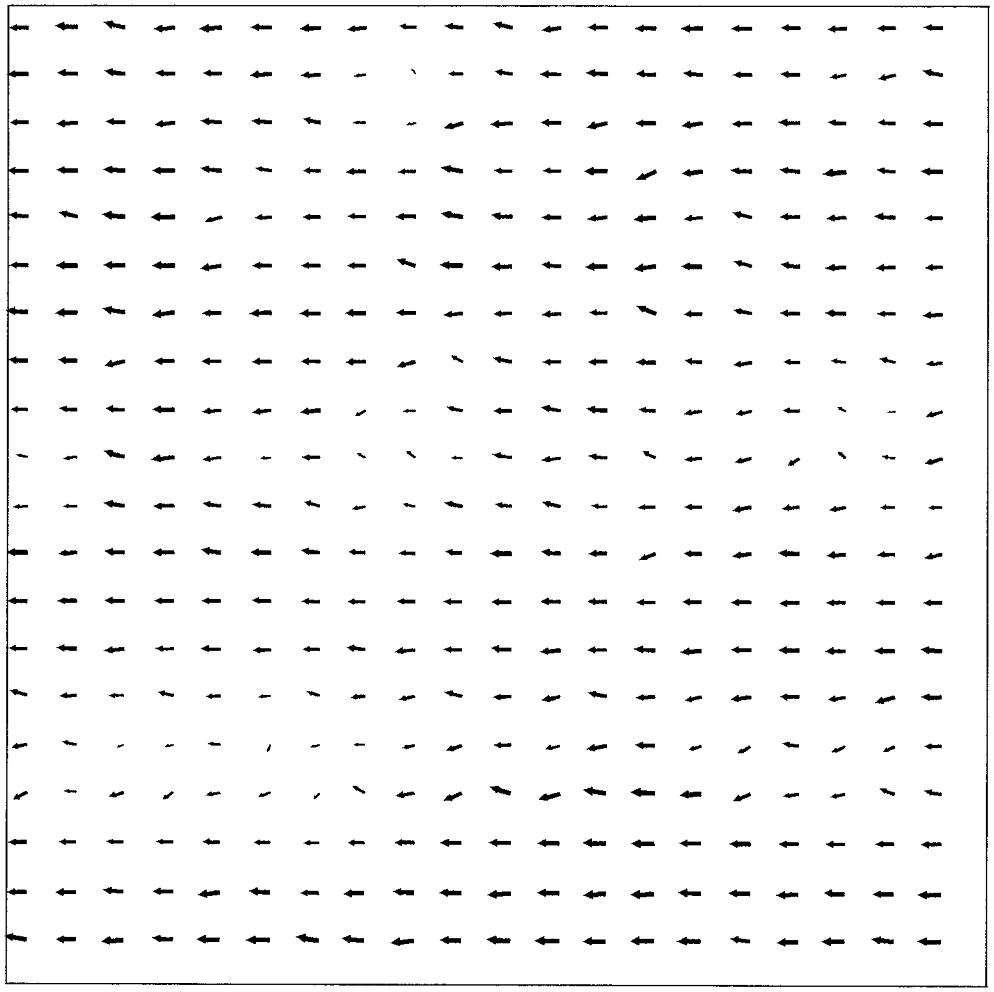


Figure 2.23 Calculated flow field in Plane Experiment (Horn)

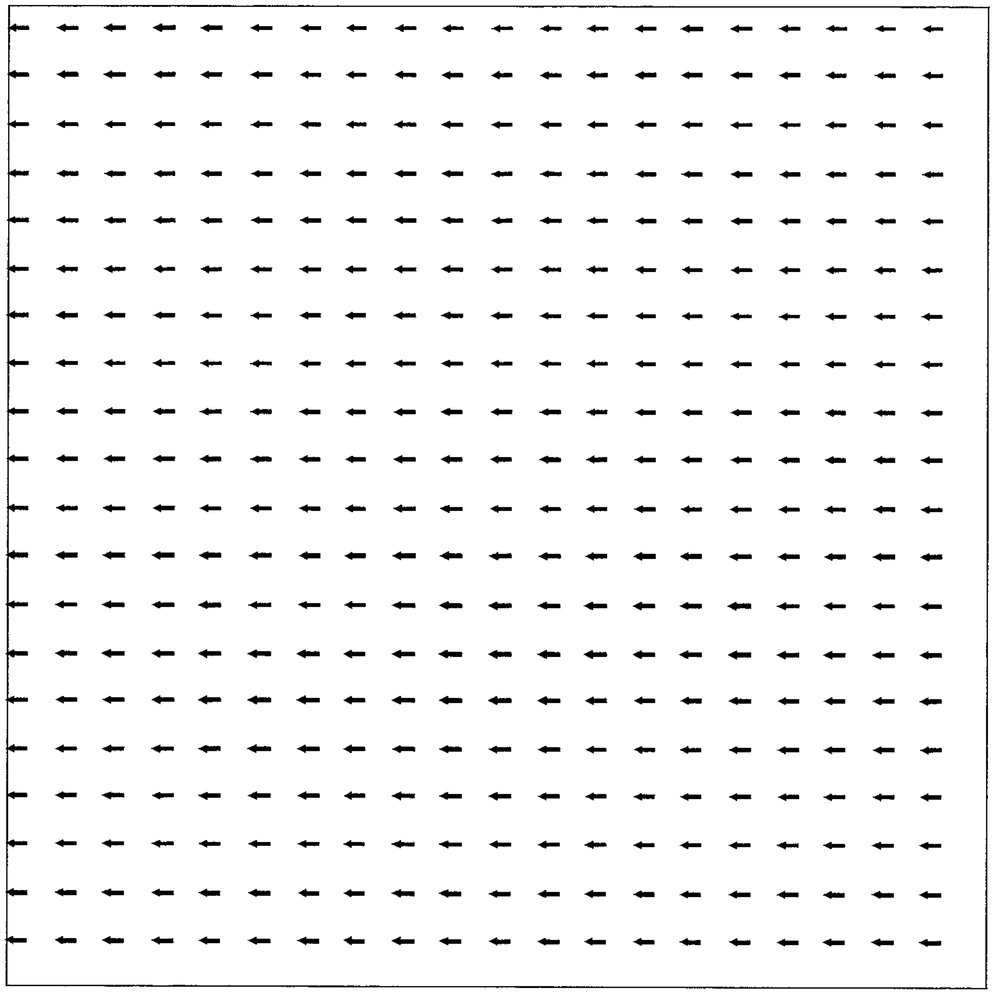


Figure 2.24 Calculated flow field in Plane Experiment (Singh)

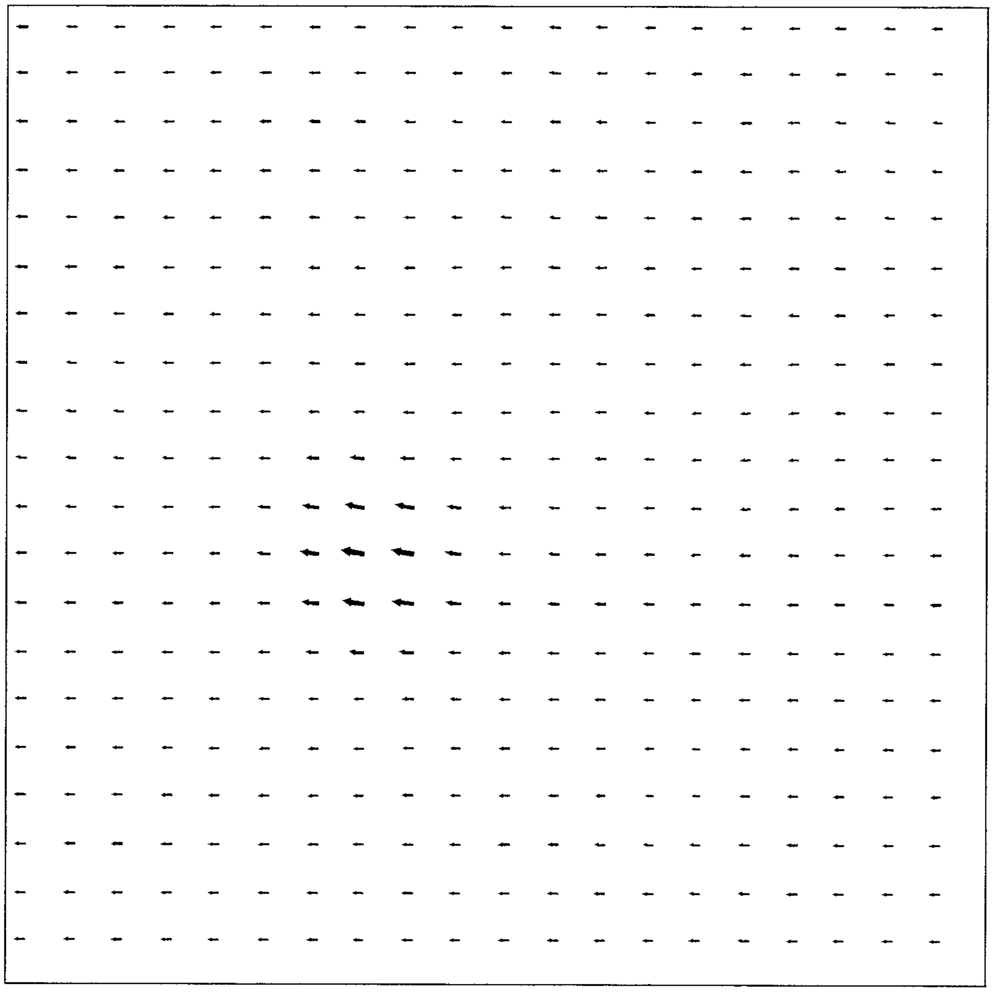


Figure 2.25 Calculated flow field in Plane Experiment (Pan)

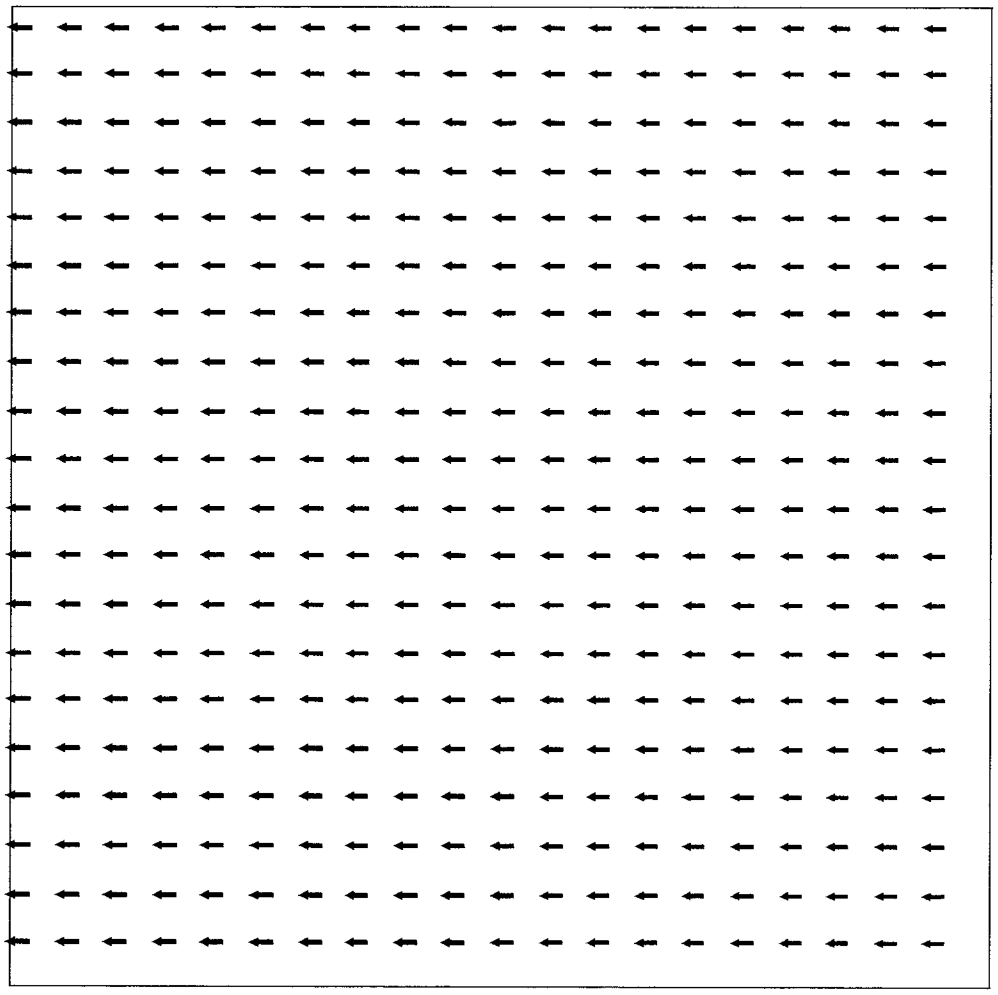


Figure 2.26 Correct flow field in Slant Experiment

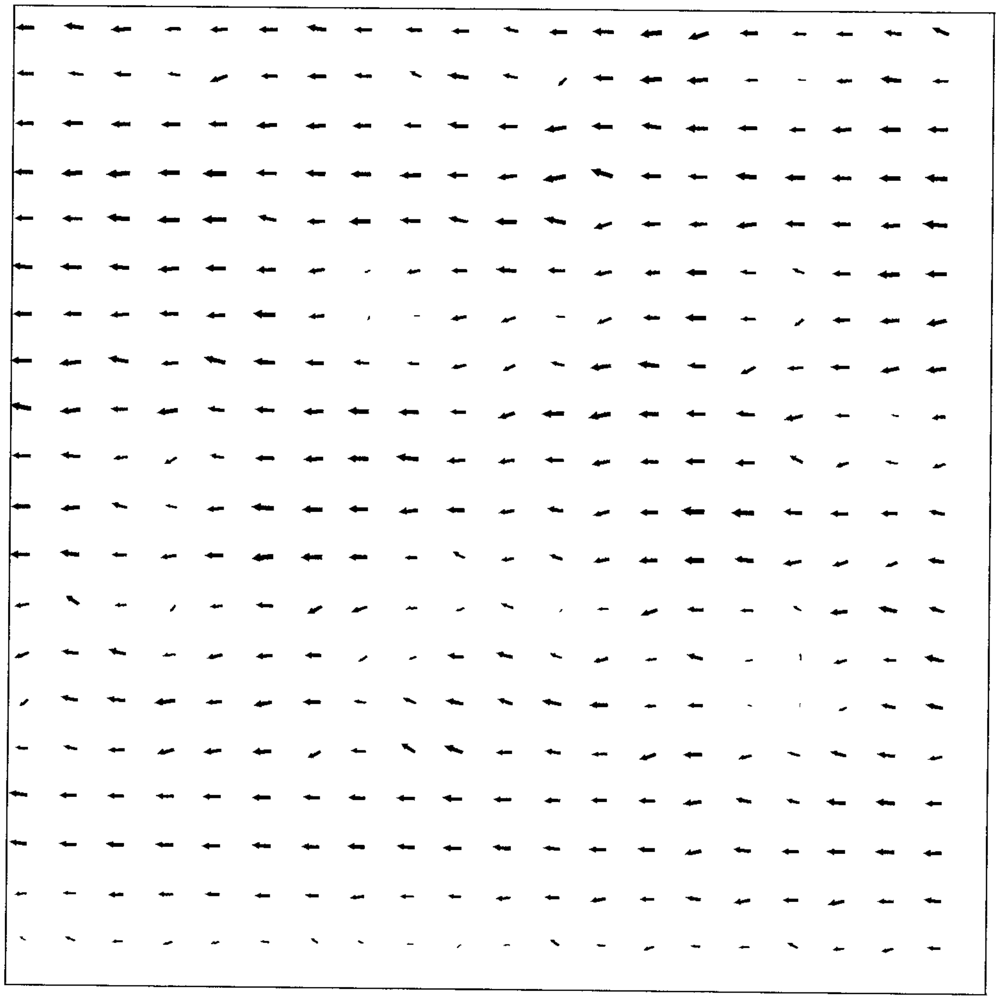


Figure 2.27 Calculated flow field in Slant Experiment (Horn)

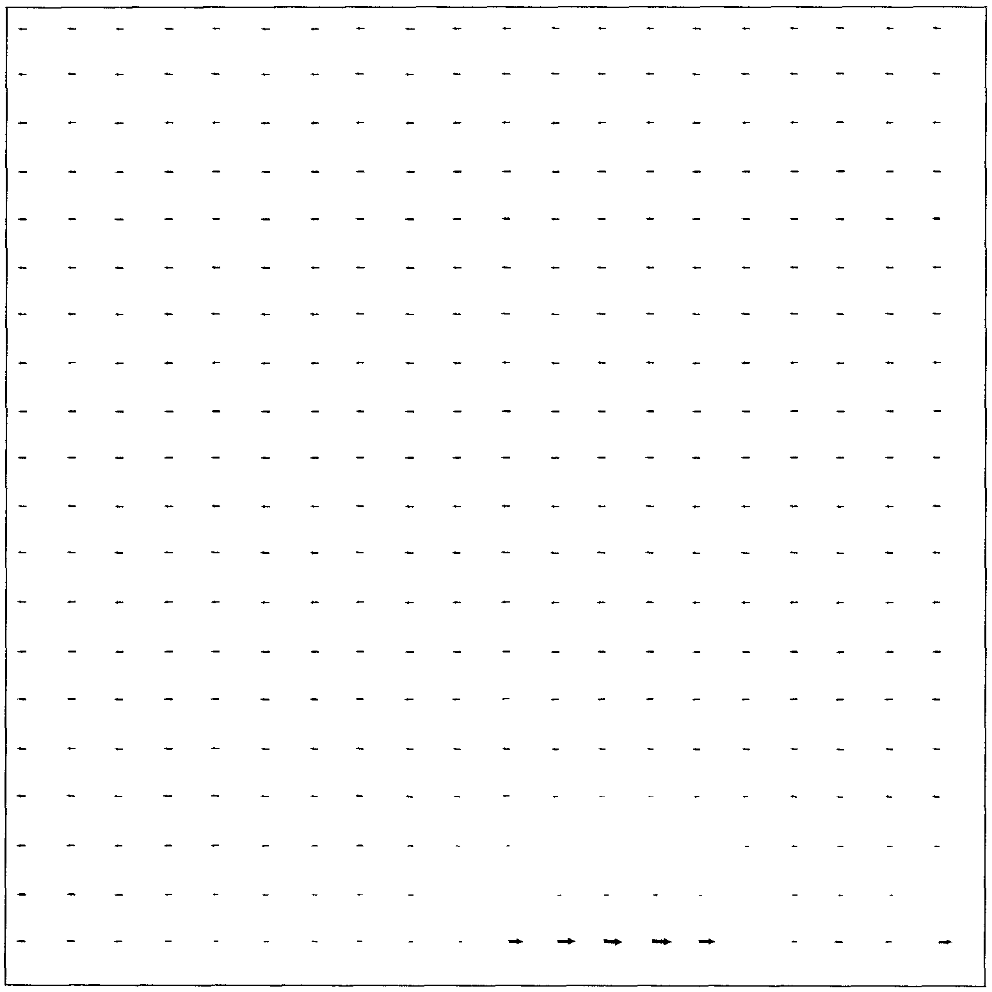


Figure 2.28 Calculated flow field in Slant Experiment (Singh)

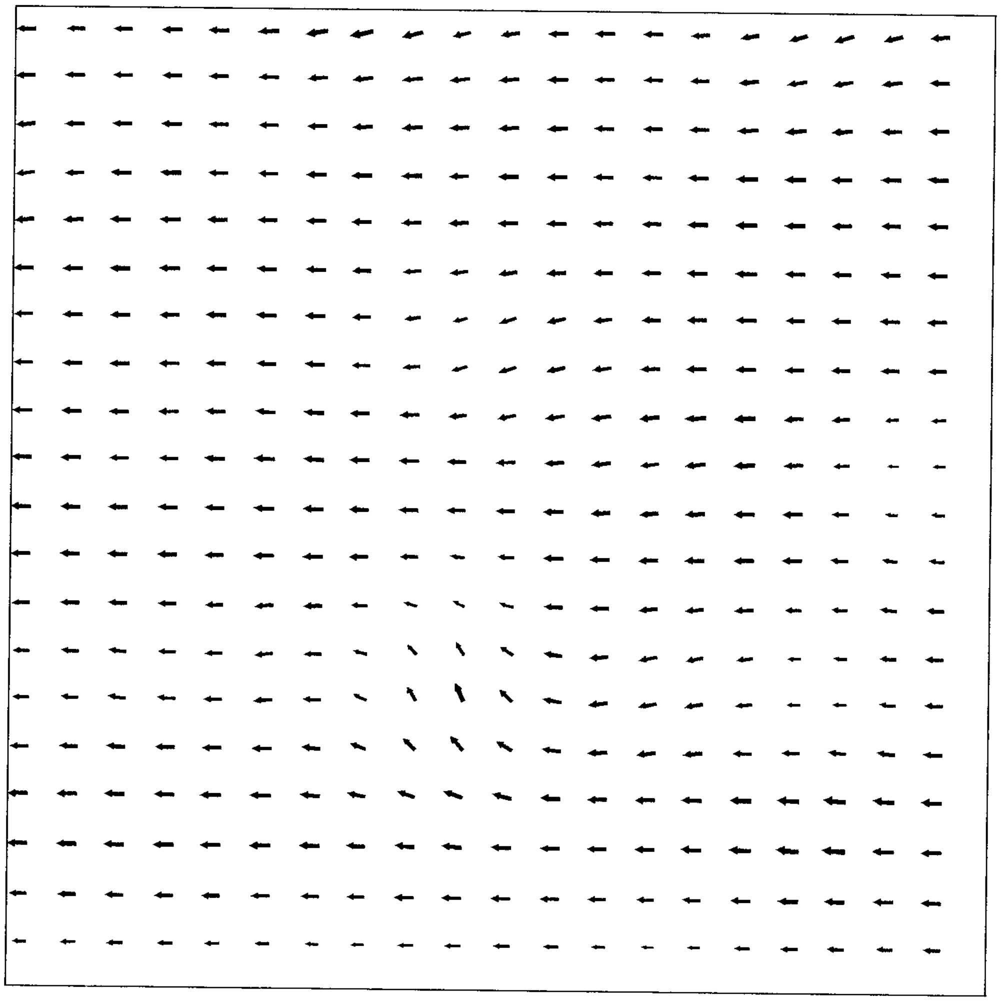


Figure 2.29 Calculated flow field in Slant Experiment (Pan)

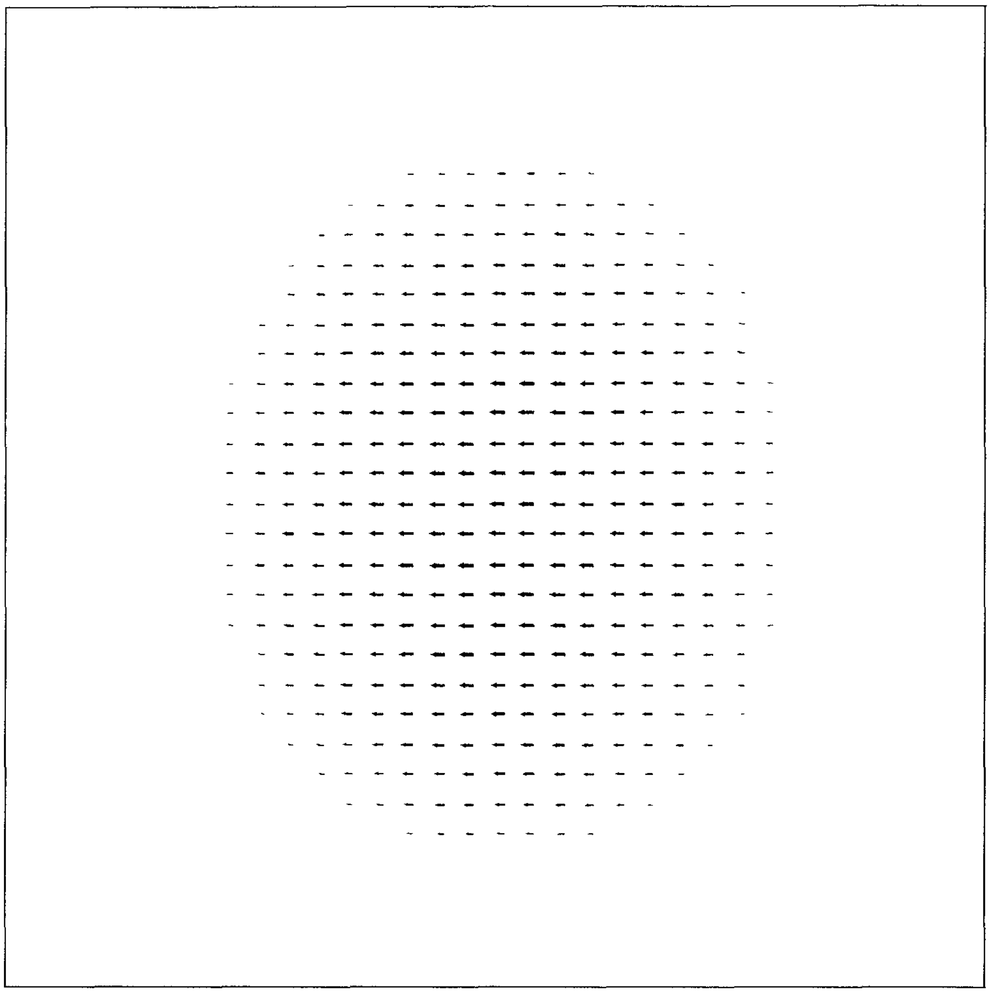


Figure 2.30 Correct flow field in Sphere Experiment (64×64 pixels)

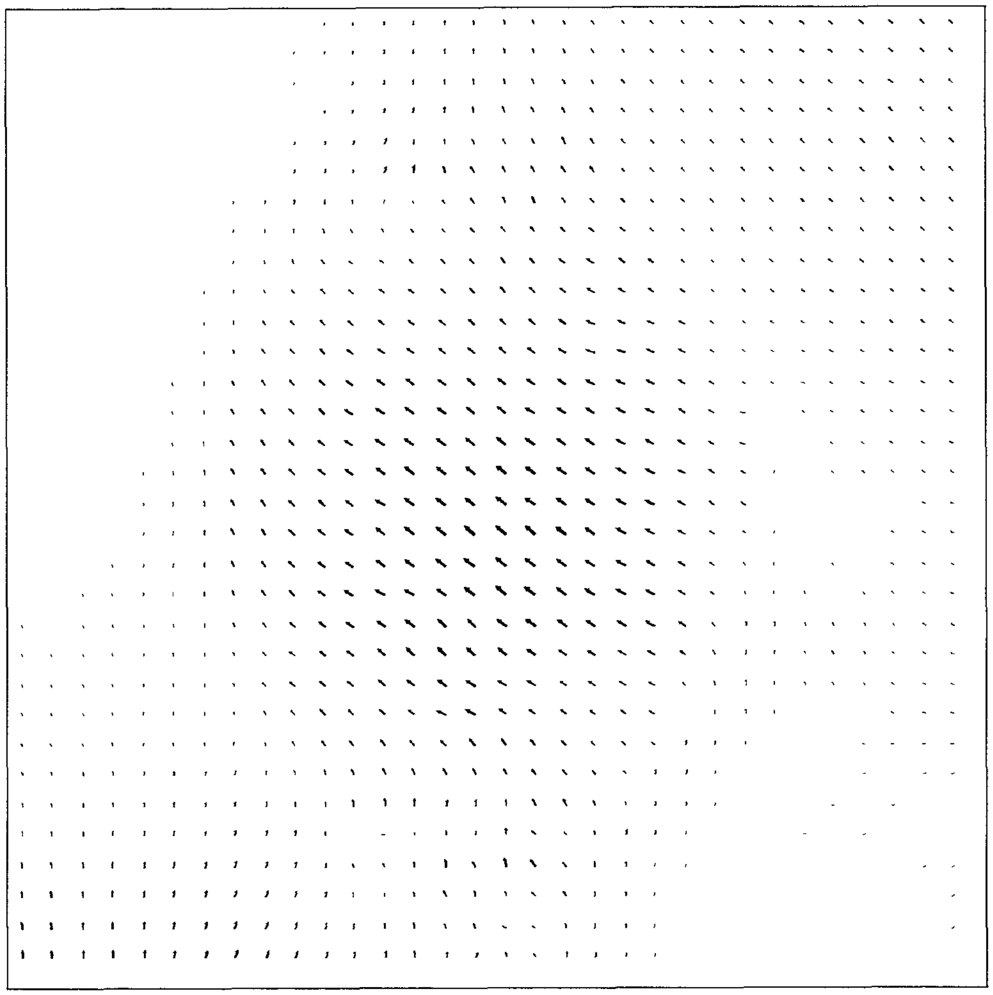


Figure 2.31 Calculated flow field in Sphere Experiment (Horn) (64×64 pixels)

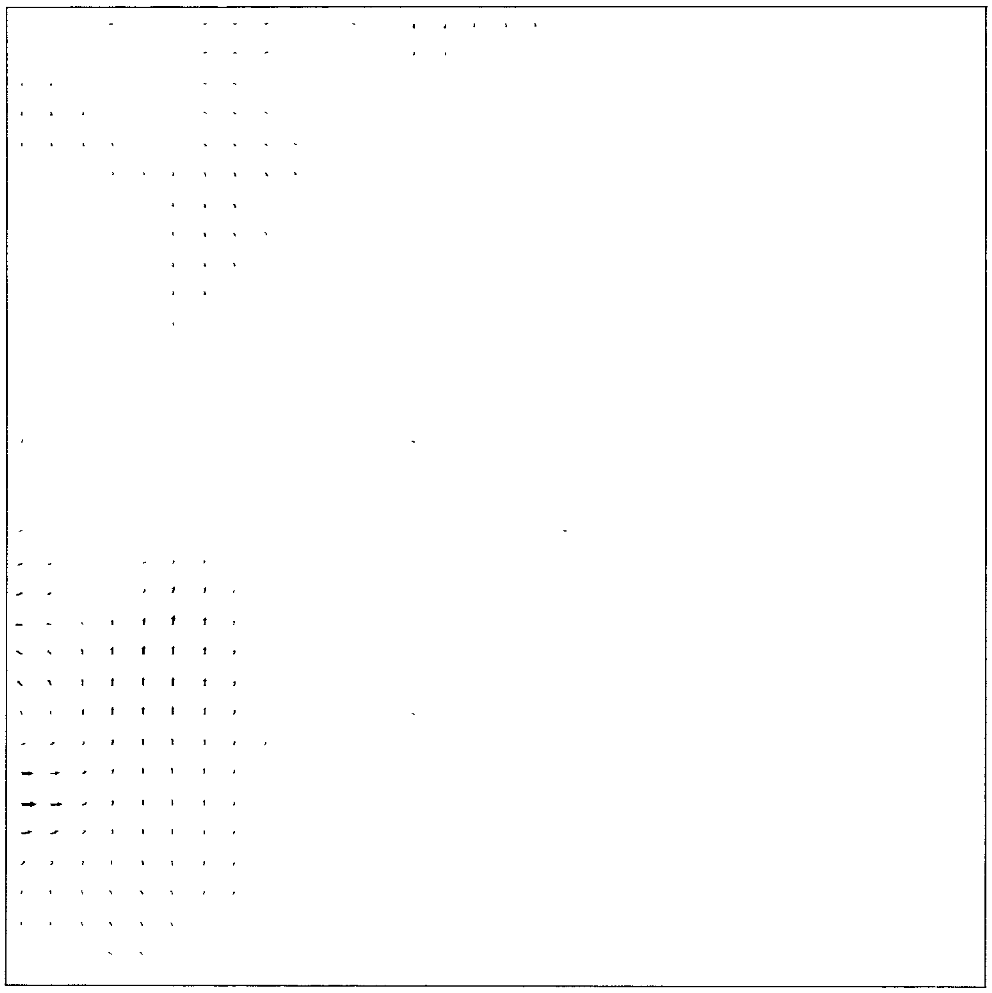


Figure 2.32 Calculated flow field in Sphere Experiment (Pan) (64×64 pixels)

CHAPTER 3

MOTION ANALYSIS

3.1 Unified Optical Flow Field (UOFF)

Over the last ten years, research shows that motion analysis has many potentials in the computer vision. In the vision system, the recovery of the motion of objects gives us the ability to estimate their speed and direction and then to navigate, recognize and track down objects. There are basically two different approaches to recover the structure of objects and the relative motion between objects and the cameras: one is the optical flow field approach and the other is the feature correspondence approach.

The feature correspondence approach requires the solution of the feature detection and the correspondence problem, which has been proven to be computationally and conceptually difficult to solve. And only partial solutions suitable for simplistic situation have been developed.

The optical flow field approach suggests us to use the distribution of apparent velocities of image brightness to recover the position and motion of the object which covers a large portion of the image. This approach achieves more robustness than feature correspondence approach at the expense of more computation in determine an optical flow. But the optical flow cannot be computed locally, and must be computed under some constraint, this results that the recovered optical flow field is not the true image flow field, so the more accurate the optical flow is calculated, the more efficient the motion analysis is.

Recently, a new approach to motion analysis from a sequence of stereo images has been proposed. It is a generalization of the optical flow to stereo imagery, it is for a whole continuous field instead of only for some features, it does not need finding feature correspondence. It is based on a Unified Optical Flow Field (UOFF) conception.

3.1.1 Imaging Space and General Brightness Function

First, consider the formation of a normal image sequence and its brightness function. Assume a sensor located in a specific position in 3-d world space, that generates images about the scene one after another. When time goes by, the images form a sequence. The set of these images can be represented by a brightness function $I(x, y, t)$, where x and y are coordinates on the image plane. This is also the basic outline of the brightness function $I(x, y, t)$ used by Horn and Schunck in the Gradient-based Approach to calculate the image flow.

Second, consider the formation of a more general image sequence and its brightness function. Assume there are many sensors instead of only one at a specific instant of time, and each sensor has its own different position to view the object in the image space, so we can not use the previous single brightness function $I(x, y, t)$ to describe these different image planes, we must use some more sophisticated mechanism to describe them, so we introduce a set of brightness function

$$I(x, y, t, \vec{s})$$

which depends not only the spatiotemporal variables x, y, t of the image plan but also the position of sensor $\vec{s}$, which contains the coordinates of the sensor center and the orientation of the optical axis of the sensor, so, $\vec{s}$ is a 5-element vector

$$\vec{s}(\hat{x}, \hat{y}, \hat{z}, \beta, \gamma)$$

where $\hat{x}$,$\hat{y}$ and $\hat{z}$ represent the coordinates of the optical center of the sensor in 3-d world space; β and γ represent the orientation of the optical axis of the sensor in 3-d world space.

More specifically, each sensor in 3-d world space can be considered associated with a 3-d Cartesian coordinate system whose center is located on the origin of the sensor and whose optical axis is aligned with the OZ axis. We choose in 3-d world space a 3-d Cartesian coordinate system as the reference coordinate system. Hence, a sensor with its associated Cartesian coordinate system coincident with the reference coordinate system, has its position in 3-d world space denoted by $\vec{s} = (0,0,0,0,0)$. An arbitrary sensor position denoted by $\vec{s} = (\hat{x}, \hat{y}, \hat{z}, \beta, \gamma)$ can be described as the following. The sensor's associated Cartesian coordinate system first has been moved from the origin of the reference coordinate system to $(\hat{x}, \hat{y}, \hat{z})$ on the reference coordinate system and then has been rotated with the rotation angles β about its OY axis, and γ about its OX axis.

The above two expressions can be used to describe the gray levels of the more general image plane.

Then, assume a point p in 3-d world space that is projected onto the image plane as a pixel with the coordinates x_p, y_p. Since the x_p, y_p are also dependent on time t and sensor's position $\vec{s}$, the coordinates of the pixel can be denoted by $x_p = x_p(t, \vec{s})$, $y_p = y_p(t, \vec{s})$. We can get

$$I = I(x_p(t, \vec{s}), x_p(t, \vec{s}), t, \vec{s}) \tag{3.1}$$

By far, we can consider the general brightness function varies with temporal variant -- time t and spatial variant -- sensor's position $\vec{s}$. It has been shown that "temporal" optical flow field $I(x_p(t, \vec{s}), x_p(t, \vec{s}), t, \vec{s} = \vec{0})$ can be computed by using the technique developed by Horn and Schunck in Gradient-based approach, and "spatial" optical flow field $I(x_p(t, \vec{s}), x_p(t, \vec{s}), t = 0, \vec{s})$ can be analyzed by using the inherent relation between a pair of stereo images obtained at the same moment. A Unified Temporal-Spatial Optical Flow

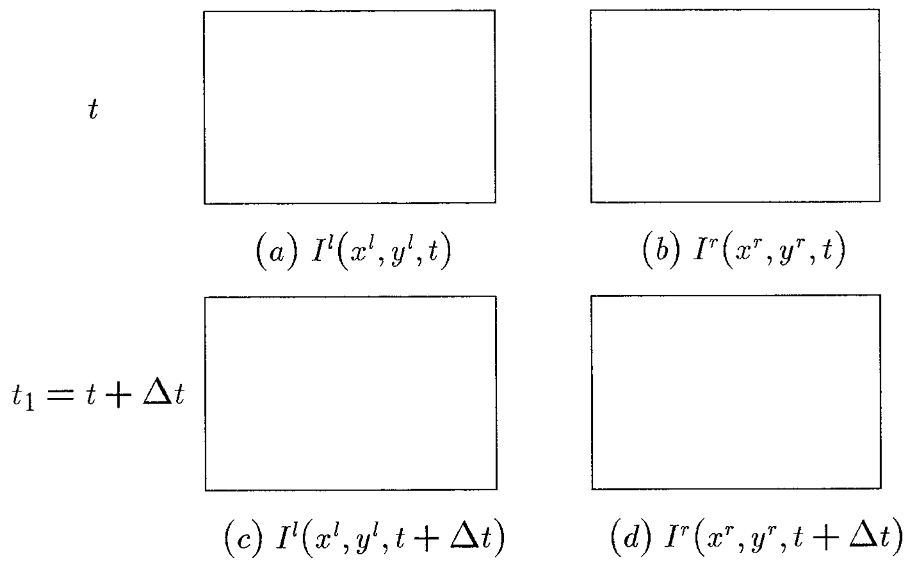


Figure 3.1 Four Frame Model

Field (UOFF) $I(x_p(t,\vec{s}), x_p(t,\vec{s}), t, \vec{s})$ is formed by combining these two types of optical flow fields. The word ''temporal-spatial'' is omitted thereafter for brevity.

3.1.2 A Four-Frame Model for UOFF

From the definition of UOFF, we know that the unified optical flow field has two variants -- time t and sensor's center position $\vec{s}$. Here we describe the procedure to obtain one specific unified optical flow field based on the four-frame model.

Figure 3.1 shows the four frame model. The four images shown here are chosen from a stereo image sequence where image (a) and (c) are taken by the left camera at moment t and $t_1 = t + \Delta t$, image (b) and (d) are taken by the right camera at time t and $t_1 = t + \Delta t$, respectively.

The following is the explanation of how to establish the unified optical flow field.

- Consider the images (a) and (c)

  These are the two images taken from the monocular image sequence generated by using the left camera. Consult the method of Horn and Schunck, we can get the following two equations:

$$(I_x^l)^2 u^l + I_x^l I_y^l v^l = \alpha_1^2 \nabla^2 u^l - I_x^l I_t^l \tag{3.2}$$

$$I_x^l I_y^l u^l + (I_y^l)^2 v^l = \alpha_1^2 \nabla^2 v^l - I_y^l I_t^l \tag{3.3}$$

  where $u^l \triangleq \lim_{\delta t \to 0} \frac{\delta x^l}{\delta t} = \frac{dx^l}{dt}$, the velocity of the pixel (x^l, y^l) along the x^l direction, and $v^l \triangleq \lim_{\delta t \to 0} \frac{\delta y^l}{\delta t} = \frac{dy^l}{dt}$, the velocity of the pixel (x^l, y^l) along the y^l direction. $I_x^l \triangleq \frac{\partial I^l(x^l,y^l,t)}{\partial x^l}$, $I_y^l \triangleq \frac{\partial I^l(x^l,y^l,t)}{\partial y^l}$, $I_t^l \triangleq \frac{\partial I^l(x^l,y^l,t)}{\partial t}$ are the partial derivatives of $I^l(x^l, y^l, t)$ with respect to x^l,y^l and t, respectively; $\nabla^2 u^l \triangleq \frac{\partial^2 u^l}{(\partial x^l)^2} + \frac{\partial^2 u^l}{(\partial y^l)^2}$ and $\nabla^2 v^l \triangleq \frac{\partial^2 v^l}{(\partial x^l)^2} + \frac{\partial^2 v^l}{(\partial y^l)^2}$ are the of u^l and v^l, respectively.

  The smoothness constraint can be used in the derivation of the above two equations and the first order derivative of $I^l(x^l, y^l, t)$, i.e., I_x^l, I_y^l, I_t^l can be determined from image data $I^l(x^l, y^l, t)$ and $I^l(x^l, y^l, t + \Delta t)$. Therefore, u^l and v^l can be solved from the above two equations.

- Consider the images (b) and (d)

  These are two images taken from the monocular image sequence generated by using the right camera. Similarly, we can get the following two equations about u^r and v^r:

$$(I_x^r)^2 u^r + I_x^r I_y^r v^r = \alpha_1^2 \nabla^2 u^r - I_x^r I_t^r \tag{3.4}$$

$$I_x^r I_y^r u^r + (I_y^r)^2 v^r = \alpha_1^2 \nabla^2 v^r - I_y^r I_t^r \tag{3.5}$$

  where $u^r, v^r, I_x^r, I_y^r, I_t^r$,$\nabla^2 u^r$ and $\nabla^2 v^r$ are the counterparts of u^l, v^l, I_x^l, I_y^l, I_t^l, $\nabla^2 u^l$ and $\nabla^2 v^l$ defined in the above step respectively. The comments

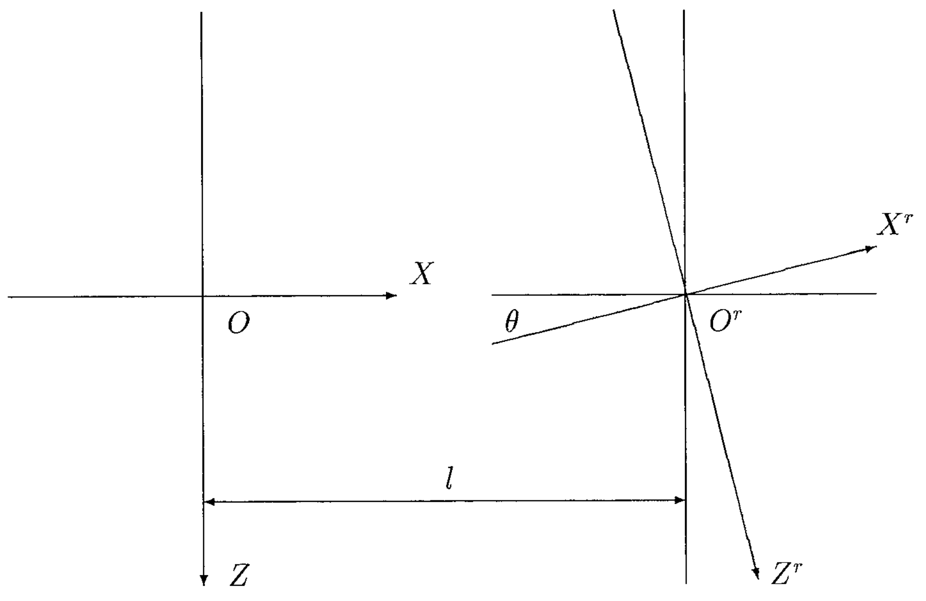


Figure 3.2 Image Geometry of Spatial Sequence

made in the above step are also applicable here. Similarly, u^r and v^r can be solved from the above two equations.

Up to now, we can compute two sets of temporal optical flow velocities, one is u^l and v^l associated with the left monocular image sequence, another is u^r and v^r associated with the right monocular image sequence.

- Consider the images (a) and (b)

These are the two images from the bicular image sequence associated with left and right camera. These images can be viewed as a ''spatial'' sequence of images with the moment ''fixed'' in the time domain.

Before considering a ''spatial'' sequence of images, the different positions of cameras in space at a specific moment should be predicted, we don't want to use too sophisticated mathematics to describe the camera's movement, so the left camera is fixed and only the right camera moves in this case, Figure 3.2 shows that.

From the Figure 3.2, the movement of the right camera can be viewed as the translation of the lens center O^r following by the rotation of the optical axis O^rZ^r. The rotation of O^rZ^r is about O^rY^r. The displacement of O^r along OX and OZ direction are denoted by $\hat{X}$ and $\hat{Z}$, respectively. The angle displacement of O^rZ^r about O^rY^r is denoted by $\hat{\theta}$. However, $\hat{Z} = 0$ when that O^r lies on OX is assumed. Define

$$\begin{aligned} \delta s &\triangleq \sqrt{\hat{X}^2 + \chi^2\hat{\theta}^2} \\ &\triangleq s^r - s^l \end{aligned}$$

-- χ is a characteristic length chosen according to image setting.

-- δs is a measurement of the variant of the right camera's position with respect to the left camera's position, i.e., the variation of the position of the right lens center O^r with respect to that of the left lens center O and the orientation of the right optical axis O^rZ^r with respect to that of the left optical axis OZ.

-- s^l is the left camera's position.

-- s^r is the right camera's position.

Also define

$$\delta x \triangleq x^r - x^l$$

$$\delta y \triangleq y^r - y^l$$

-- δx is the horizontal coordinate difference of the image point on the right and left image plane, corresponding to the same point in 3-d world space.

-- δy is the vertical coordinate difference of the image point on the right and left image plane, corresponding to the same point in 3-d world space.

Corresponding to the same point in 3-d world space, the invariance of the brightness of a pair of image pixel can be illustrated by the following equation

$$I^s(x^l, y^l, t)|_{s=s^l} = I^s(x^r, y^r, t)|_{s=s^r} \tag{3.6}$$

Since $\delta x \triangleq x^r - x^l$ and $\delta y \triangleq y^r - y^l$, and using Taylor's series, the following equations can be obtained

$$\begin{aligned} I^s(x^r, y^r, t)|_{s=s^r} &= I^s(x^l + \delta x, y^l + \delta y, t)|_{s=s^l+\delta s} \\ &= I^s(x^l, y^l, t)|_{s=s^l} + \frac{\partial I^l}{\partial x}\delta x + \frac{\partial I^l}{\partial y}\delta y + \frac{\partial I^l}{\partial s}\delta s + \varepsilon \end{aligned}$$

Following the above steps, it shows that

$$\frac{\partial I^l}{\partial x}\delta x + \frac{\partial I^l}{\partial y}\delta y + \frac{\partial I^l}{\partial s}\delta s + \varepsilon = 0 \tag{3.7}$$

Dividing the both sides of the above equation by δs, and let $\delta s \to 0$, considering the fact that ε contains the second and higher order terms of δx,δy and δs, we can ignore the term $\varepsilon/\delta s$ when $\delta s \to 0$, this gives

$$\frac{\partial I^l}{\partial x}u^s + \frac{\partial I^l}{\partial y}v^s + \frac{\partial I^l}{\partial s} = 0 \tag{3.8}$$

where

$$u^s \triangleq \lim_{\delta s \to 0} \frac{\delta x}{\delta s}$$

$$v^s \triangleq \lim_{\delta s \to 0} \frac{\delta y}{\delta s}$$

$$\begin{aligned} \frac{\partial I^l}{\partial s} &\triangleq \lim_{\delta s \to 0} \frac{I^s(x^l, y^l, t)|_{s=s^r} - I^s(x^l, y^l, t)|_{s=s^l}}{\delta s} \\ &\approx \frac{I^r(x^l, y^l, t) - I^l(x^l, y^l, t)}{\delta s} \end{aligned}$$

Using the same method which is used in Horn and Schunck's approach, the following equations can be derived to solve the two unknowns u^s and v^s

$$(I^l_x)^2 u^s + I^l_x I^l_y v^s = \alpha_2^2 \nabla^2 u^s - I^l_x I^l_s \tag{3.9}$$

$$I_x^l I_y^l u^s + (I_y^l)^2 v^s = \alpha_2^2 \nabla^2 v^r - I_y^l I_s^l \tag{3.10}$$

- Consider the images (a), (b), (c) and (d).

So far, an unified optical flow field can be established with consisting of the following six field quantities: u^l,v^l,u^r,v^r,u^s,v^s which can be solve from the following equations (which are just conclusion from the above steps)

$$\begin{aligned}
(I_x^l)^2 u^l + I_x^l I_y^l v^l &= \alpha_1^2 \nabla^2 u^l - I_x^l I_t^l && (3.11)\\
I_x^l I_y^l u^l + (I_y^l)^2 v^l &= \alpha_1^2 \nabla^2 v^l - I_y^l I_t^l && (3.12)\\
(I_x^r)^2 u^r + I_x^r I_y^r v^r &= \alpha_1^2 \nabla^2 u^r - I_x^r I_t^r && (3.13)\\
I_x^r I_y^r u^r + (I_y^r)^2 v^r &= \alpha_1^2 \nabla^2 v^r - I_y^r I_t^r && (3.14)\\
(I_x^l)^2 u^s + I_x^l I_y^l v^s &= \alpha_2^2 \nabla^2 u^s - I_x^l I_s^l && (3.15)\\
I_x^l I_y^l u^s + (I_y^l)^2 v^s &= \alpha_2^2 \nabla^2 v^s - I_y^l I_s^l && (3.16)
\end{aligned}$$

By using the calculus of variation and using the approximation to the Laplacian, linear equations can be derived from the above equations for variables $u^l, v^l, u^r, v^r, u^s, v^s$, solving these linear equations, we can find that

$$(\alpha_1^2 + (I_x^l)^2 + (I_y^l)^2) u^l = +(\alpha_1^2 + (I_y^l)^2)\overline{u^l} - I_x^l I_y^l \overline{v^l} - I_x^l I_t^l \tag{3.17}$$

$$(\alpha_1^2 + (I_x^l)^2 + (I_y^l)^2) v^l = -I_x^l I_y^l \overline{u^l} + (\alpha_1^2 + (I_x^l)^2)\overline{v^l} - I_y^l I_t^l \tag{3.18}$$

$$(\alpha_1^2 + (I_x^r)^2 + (I_y^r)^2) u^r = +(\alpha_1^2 + (I_y^r)^2)\overline{u^r} - I_x^r I_y^r \overline{v^r} - I_x^r I_t^r \tag{3.19}$$

$$(\alpha_1^2 + (I_x^r)^2 + (I_y^r)^2) v^r = -I_x^r I_y^r \overline{u^r} + (\alpha_1^2 + (I_x^r)^2)\overline{v^r} - I_y^r I_t^r \tag{3.20}$$

$$(\alpha_1^2 + (I_x^s)^2 + (I_y^s)^2) u^s = +(\alpha_1^2 + (I_y^s)^2)\overline{u^s} - I_x^s I_y^s \overline{v^s} - I_x^s I_t^s \tag{3.21}$$

$$(\alpha_1^2 + (I_x^s)^2 + (I_y^s)^2) v^s = -I_x^s I_y^s \overline{u^s} + (\alpha_1^2 + (I_x^s)^2)\overline{v^s} - I_y^s I_t^s \tag{3.22}$$

Finally, image velocities $u^l, v^l, u^r, v^r, u^s, v^s$ can be computed with iterative procedure

$$(u^l)^{n+1} = (\overline{u^l})^n - I_x^l \frac{I_x^l(\overline{u^l})^n + I_y^l(\overline{v^l})^n + I_t^l}{\alpha_1^2 + (I_x^l)^2 + (I_y^l)^2} \tag{3.23}$$

$$(v^l)^{n+1} = (\overline{v^l})^n - I_y^l \frac{I_x^l(\overline{u^l})^n + I_y^l(\overline{v^l})^n + I_t^l}{\alpha_1^2 + (I_x^l)^2 + (I_y^l)^2} \tag{3.24}$$

$$(u^r)^{n+1} = (\overline{u^r})^n - I_x^r \frac{I_x^r(\overline{u^r})^n + I_y^r(\overline{v^r})^n + I_t^r}{\alpha_1^2 + (I_x^r)^2 + (I_y^r)^2} \tag{3.25}$$

$$(v^r)^{n+1} = (\overline{v^r})^n - I_y^r \frac{I_x^r(\overline{u^r})^n + I_y^r(\overline{v^r})^n + I_t^r}{\alpha_1^2 + (I_x^r)^2 + (I_y^r)^2} \tag{3.26}$$

$$(u^s)^{n+1} = (\overline{u^s})^n - I_x^s \frac{I_x^s(\overline{u^s})^n + I_y^s(\overline{v^s})^n + I_t^s}{\alpha_1^2 + (I_x^s)^2 + (I_y^s)^2} \tag{3.27}$$

$$(v^s)^{n+1} = (\overline{v^s})^n - I_y^s \frac{I_x^s(\overline{u^s})^n + I_y^s(\overline{v^s})^n + I_t^s}{\alpha_1^2 + (I_x^s)^2 + (I_y^s)^2} \tag{3.28}$$

3.2 Equations for Analysis of 3-d Motion Field Based on UOFF

In this section, it will be shown that when the six UOFF quantities u^l,v^l,u^r,v^r,u^s,v^s are given, the six motion field quantities X,Y,Z,$\not{X}$,$\not{Y}$,$\not{Z}$ are easy to be obtained. That also means the six field quantities u^l,v^l,u^r,v^r,u^s,v^s contain enough information to recover the motion in 3-d world space. It's obvious that x^l,y^l,x^r,y^r are the input data for the input image sequence.

In this thesis, two cases of object movement are considered: one is the object moving around corresponding to the optical axis of the fixed left camera, the other is the object moving normal to the optical axis of the fixed left camera.

3.2.1 Imaging Geometry 1: Rotation

It should be noted that the six UOFF quantities u^l,v^l,u^r,v^r,u^s,v^s are known, the six motion field quantities X,Y,Z,$\not{X}$,$\not{Y}$,$\not{Z}$ are unknown.

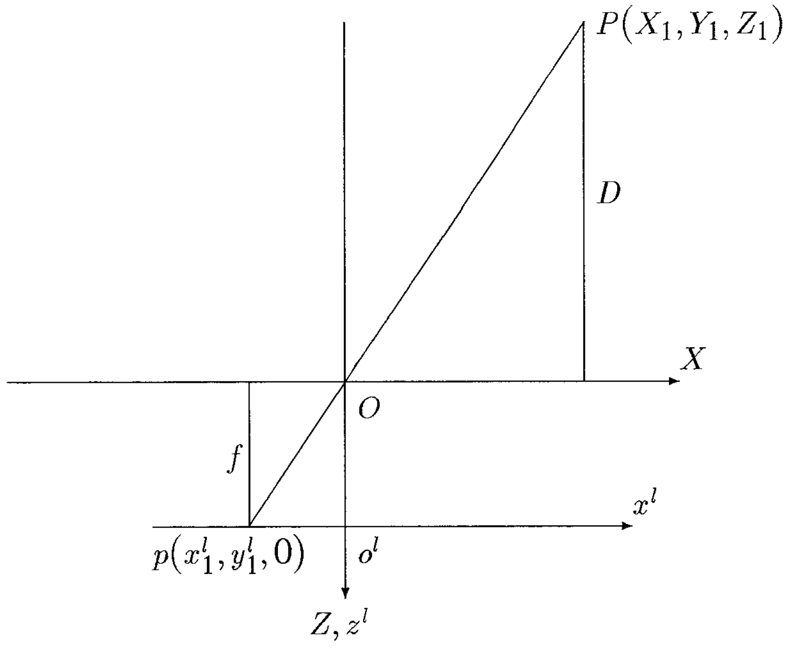


Figure 3.3 The Perspective Projection of The Left Camera

About rotation moving, the image geometry can be considered as Figure 3.2. Assuming a point in 3-d world space, its coordinates in the two coordinate systems are (X,Y,Z) and (X^r,Y^r,Z^r) which have the following relation

$$\begin{pmatrix} X^r \\ Y^r \\ Z^r \end{pmatrix} = R \times \left(\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + T \right) \tag{3.29}$$

where R is a rotation matrix and T is a translation vector.

$$R = \begin{pmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{pmatrix} \quad T = \begin{pmatrix} -l \\ 0 \\ 0 \end{pmatrix}$$

Figure 3.3 shows the perspective projection plane of the left camera. Before any further discussion, one assumption should be remembered that the discussed world point is far away from the camera in the whole thesis.

Following from the geometry optics, the relation between the world point $P(X,Y,Z)$ and its image point $p_l(x,y)$ is that

$$x^l = \frac{f^l}{Z}X \qquad y^l = \frac{f^l}{Z}Y \tag{3.30}$$

Similarly, the relation between the world point $P(X, Y, Z)$ and its image point $p_r(x, y)$ is that

$$x^r = \frac{f^r}{Z^r} X^r \qquad y^r = \frac{f^r}{Z^r} Y^r \tag{3.31}$$

Assuming $f^l = f^r = f$ in this thesis, and using the relation between (X, Y, Z) and (X^r, Y^r, Z^r), the following results can be gotten

$$x^l = \frac{f}{Z} X \tag{3.32}$$

$$y^l = \frac{f}{Z} Y \tag{3.33}$$

$$\begin{aligned} x^r &= \frac{f}{Z^r} X^r \\ &= f \frac{(X - l)cos\theta - Zsin\theta}{Xsin\theta + Zcos\theta} \\ &\quad (the\ far\ field\ assumption\ Z^r \approx Z) \\ &\approx \frac{f}{Z}((X - l)cos\theta - Zsin\theta) \end{aligned} \tag{3.34}$$

$$\begin{aligned} y^r &= \frac{f}{Z^r} Y^r \\ &\quad (condition\ Y^r = Y and\ Z^r \approx Z) \\ &\approx \frac{f}{Z} Y \end{aligned} \tag{3.35}$$

The relative relation between X and Z can be derived from $Z \approx Z^r$, it can be proved as the following:

$$\begin{aligned} Z &\approx Z^r \\ &= Zcos\theta + Xsin\theta \\ &\quad (divided\ by\ Z\ on\ both\ sides) \\ 1 &\approx cos\theta + \frac{X}{Z} sin\theta \\ &\quad (do\ calculation\ on\ both\ sides) \\ \frac{X}{Z} &\approx \frac{1 - cos\theta}{sin\theta} \end{aligned} \tag{3.36}$$

Following the definition of $u^s = \lim_{\delta s \to 0} \frac{\delta x}{\delta s}$

$$\begin{aligned}
\frac{u^s}{f} &= \frac{1}{f}\lim_{\delta s \to 0}\frac{\delta x}{\delta s} \approx \frac{1}{f}\frac{\delta x}{\delta s} \\
&= \frac{x^r - x^l}{f}\frac{1}{\delta s} \\
&= (\frac{X^r}{Z^r} - \frac{X^l}{Z^l})\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
&= (\frac{(X^l - l)cos\theta - Z^l sin\theta}{Z^r} - \frac{X^l}{Z^l})\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
& \quad (assumption\ Z^r \approx Z,\ X^l = X,\ Z^l = Z) \\
&\approx (\frac{(X-l)cos\theta - Z sin\theta}{Z} - \frac{X}{Z})\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
&= (\frac{X cos\theta - l sin\theta - Z sin\theta - X}{Z})\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
&= (\frac{X cos\theta - X}{Z} - \frac{l cos\theta}{Z} - sin\theta)\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
& \quad (using\ \frac{X}{Z} \approx \frac{1-cos\theta}{sin\theta}) \\
&\approx -(\frac{l cos\theta}{Z} + sin\theta + \frac{1-cos\theta}{sin\theta}(1-cos\theta))\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
&= -(\frac{l cos\theta}{Z} + \frac{(sin\theta)^2 + 1 - 2cos\theta + (cos\theta)^2}{sin\theta})\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
&= -(\frac{l cos\theta}{Z} + \frac{2(1-cos\theta)}{sin\theta})\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \qquad (3.37)
\end{aligned}$$

Up to now, Z can be solved from the above equation, the procedure is

$$\begin{aligned}
\frac{u^s}{f} &\approx -(\frac{l cos\theta}{Z} + \frac{2(1-cos\theta)}{sin\theta})\frac{1}{\sqrt{l^2+\chi^2\theta^2}} \\
\frac{l cos\theta}{Z} &\approx -\frac{u^s\sqrt{l^2+\chi^2\theta^2}}{f} - \frac{2(1-cos\theta)}{sin\theta} \\
&= -\frac{u^s sin\theta\sqrt{l^2+\chi^2\theta^2} + 2f(1-cos\theta)}{f sin\theta} \qquad (3.38)
\end{aligned}$$

The final result about Z can be computed as following

$$Z = -\frac{f\, l\, sin\,\theta\, cos\,\theta}{u^s\, sin\,\theta\sqrt{l^2+\chi^2\theta^2} + 2f\,(1 - cos\,\theta)} \qquad (3.39)$$

From the previous derivation, we have

$$X = \frac{x^l Z}{f} \tag{3.40}$$

$$Y = \frac{y^l Z}{f} \tag{3.41}$$

Following the definition of u^l and u^r,

$$u^l \triangleq \lim_{\delta t \to 0} \frac{\delta x^l}{\delta t} = \frac{dx^l}{dt}$$

$$u^r \triangleq \lim_{\delta t \to 0} \frac{\delta x^r}{\delta t} = \frac{dx^r}{dt}$$

It's obvious that

$$\begin{aligned}
\frac{1}{f}u^l &= \frac{1}{f}\frac{dx^l}{dt} = \frac{d(\frac{x^l}{f})}{dt} = \frac{d(\frac{X^l}{Z^l})}{dt} \\
&= \frac{d(\frac{X}{Z})}{dt} \\
&= \frac{\dot{X}Z - X\dot{Z}}{Z^2}
\end{aligned} \tag{3.42}$$

$$\begin{aligned}
\frac{1}{f}u^r &= \frac{1}{f}\frac{dx^r}{dt} = \frac{d(\frac{x^r}{f})}{dt} = \frac{d(\frac{X^r}{Z^r})}{dt} \\
&\approx \frac{d(\frac{(X-l)cos\theta - Zsin\theta}{Z})}{dt} \\
&= \frac{(\dot{X}cos\theta - \dot{Z}sin\theta)Z - ((X-l)cos\theta - Zsin\theta)\dot{Z}}{Z^2} \\
&= \frac{\dot{X}Zcos\theta - \dot{Z}Zsin\theta - X\dot{Z}cos\theta + l\dot{Z}cos\theta + \dot{Z}Zsin\theta}{Z^2} \\
&= \frac{\dot{X}Zcos\theta - X\dot{Z}cos\theta + l\dot{Z}cos\theta}{Z^2} \\
&= \frac{\dot{X}Z - X\dot{Z}}{Z^2}cos\theta + \frac{l\dot{Z}}{Z^2}cos\theta \\
&\quad (using\ the\ equation\ 3.42) \\
&= \frac{1}{f}u^l cos\theta + \frac{l\dot{Z}}{Z^2}cos\theta
\end{aligned} \tag{3.43}$$

From now on, the $\dot{Z}$ can be solved from the above equation 3.43

$$\frac{l\dot{Z}}{Z^2}cos\theta = -\frac{1}{f}u^l cos\theta + \frac{1}{f}u^r$$

$$\not{Z} = \frac{Z^2(u^r - u^l cos\theta)}{flcos\theta} \tag{3.44}$$

It follows from Equation 3.42 that

$$\not{X} = \frac{X\not{Z}}{Z} + \frac{u^l Z}{f} \tag{3.45}$$

By the definition of v^l, v^r,

$$v^l \triangleq \lim_{\delta t \to 0} \frac{\delta y^l}{\delta t} = \frac{dy^l}{dt}$$
$$v^r \triangleq \lim_{\delta t \to 0} \frac{\delta y^r}{\delta t} = \frac{dy^r}{dt}$$

and using the property of this image setting $Y = Y^r$, we know that $v^r = v^l$

$$\begin{aligned} \frac{1}{f} v^l &= \frac{1}{f}\frac{dy^l}{dt} \\ &= \frac{d(\frac{Y}{Z})}{dt} \\ &= \frac{\not{Y}Z - Y\not{Z}}{Z^2} \\ &= \frac{\not{Y}}{Z} - \frac{Y\not{Z}}{Z^2} \end{aligned} \tag{3.46}$$

From Equation 3.46, $\not{Y}$ can be obtained as

$$\not{Y} = \frac{Y\not{Z}}{Z} + \frac{v^l Z}{f} = \frac{Y\not{Z}}{Z} + \frac{v^r Z}{f} \tag{3.47}$$

Here is summary equations for recovering the six 3-d motion field quantities

$$Z = -\frac{f\, l\, sin\,\theta\, cos\,\theta}{u^s\, sin\,\theta\,\sqrt{l^2 + \chi^2\,\theta^2} + 2f\,(1 - cos\,\theta)} \tag{3.48}$$

$$X = \frac{x^l\, Z}{f} \tag{3.49}$$

$$Y = \frac{y^l Z}{f} \tag{3.50}$$

$$\not{Z} = \frac{Z^2\left(u^r - u^l\, cos\,\theta\right)}{f\, l\, cos\,\theta} \tag{3.51}$$

$$\not{X} = \frac{X\not{Z}}{Z} + \frac{u^l\, Z}{f} \tag{3.52}$$

$$\not{Y} = \frac{Y\not{Z}}{Z} + \frac{v^l\, Z}{f} \tag{3.53}$$

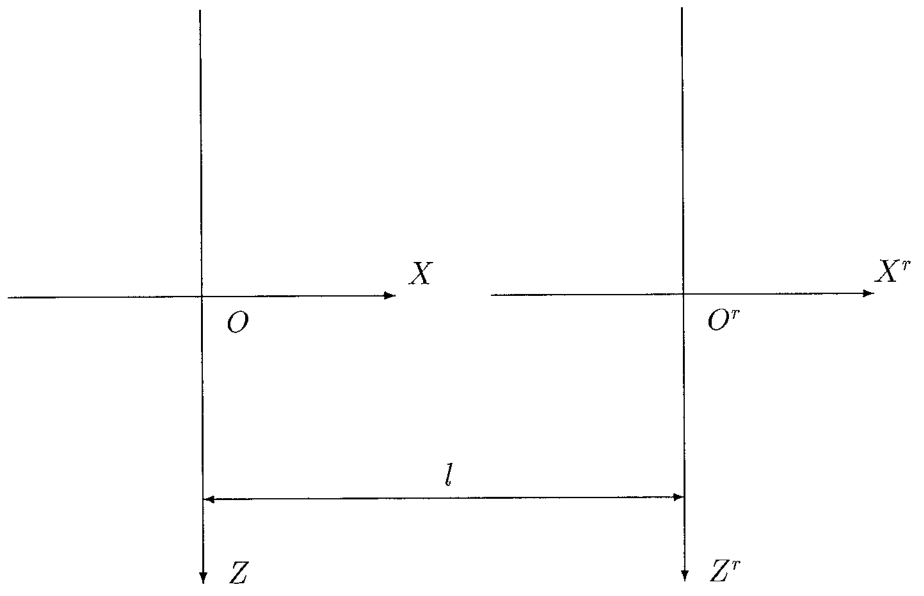


Figure 3.4 Imaging Geometry-2: Parallel

3.2.2 Imaging Geometry 2: Parallel

The parallel imaging geometry can be considered as Figure 3.4. Assuming a point in 3-d world space, its coordinates in the two coordinate systems are (X, Y, Z) and (X^r, Y^r, Z^r) which have the following relation

$$\begin{pmatrix} X^r \\ Y^r \\ Z^r \end{pmatrix} = \left(\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + T \right) \tag{3.54}$$

where T is a translation vector: $T = \begin{pmatrix} -l \\ 0 \\ 0 \end{pmatrix}$

It is worth noticing that the perspective projection of the left camera is almost the same case as the rotation imaging geometry which is shown in the Figure 3.3. The same assumption that the world point is far away from the camera. The only difference is δs , which is the measurement of the variant of the right camera's position with respect to the left camera's position.

$$\begin{aligned} \delta s &\triangleq \sqrt{\hat{X}^2 + \chi^2\hat{\theta}^2} \\ & \quad (where\ in\ this\ case\ \theta = 0) \\ &= \sqrt{(-l)^2} \;=\; |l| \end{aligned}$$

Similarly, following the definition of $u^s = \lim_{\delta s \to 0} \frac{\delta x}{\delta s}$, the below derivative can be reached

$$\begin{aligned}
\frac{u^s}{f} &= \frac{1}{f} \lim_{\delta s \to 0} \frac{\delta x}{\delta s} \\
&\approx \frac{1}{f} \frac{\delta x}{\delta s} \\
&= \frac{x^r - x^l}{f} \frac{1}{\delta s} \\
&= \left(\frac{X^r}{Z^r} - \frac{X^l}{Z^l}\right) \frac{1}{|l|} \\
& \left(Usingassumption\ Z^r \approx Z^l = Z\right) \\
&\approx \left(\frac{(X - l)}{Z} - \frac{X}{Z}\right) \frac{1}{|l|} \\
&= -\frac{l}{Z|l|} \qquad (3.55)
\end{aligned}$$

Also,following the definition of u^l and u^r,

$$u^l \triangleq \lim_{\delta t \to 0} \frac{\delta x^l}{\delta t} = \frac{dx^l}{dt}$$

$$u^r \triangleq \lim_{\delta t \to 0} \frac{\delta x^r}{\delta t} = \frac{dx^r}{dt}$$

It's obvious that

$$\begin{aligned}
\frac{1}{f} u^l &= \frac{1}{f} \frac{dx^l}{dt} = \frac{d\left(\frac{x^l}{f}\right)}{dt} \\
&= \frac{d\left(\frac{X^l}{Z^l}\right)}{dt} = \frac{d\left(\frac{X}{Z}\right)}{dt} \\
&= \frac{\underline{X} Z - X \underline{Z}}{Z^2} \qquad (3.56)
\end{aligned}$$

$$\begin{aligned}
\frac{1}{f} u^r &= \frac{1}{f} \frac{dx^r}{dt} = \frac{d\left(\frac{x^r}{f}\right)}{dt} \\
&= \frac{d\left(\frac{X^r}{Z^r}\right)}{dt} = \frac{d\left(\frac{(X-l)}{Z}\right)}{dt} \\
&= \frac{\underline{X} Z - X \underline{Z} + l \underline{Z}}{Z^2} \\
&= \frac{\underline{X} Z - X \underline{Z}}{Z^2} + \frac{l \underline{Z}}{Z^2}
\end{aligned}$$

$$\left(\textit{using the equation } 3.56\right)$$
$$= \frac{1}{f}u^l + \frac{l\not{Z}}{Z^2} \qquad (3.57)$$

From now on, the $\not{Z}$ can be solved from the above equation

$$\not{Z} = \frac{Z^2(u^r - u^l)}{f\,l} \qquad (3.58)$$

Referring to the previous section to get more detail for the derivatives, the following six equations can be derived to solve the six unknown motion field quantities

$$Z = -\frac{f\,l}{u^s\,|\,l\,|} \qquad (3.59)$$
$$X = \frac{x^l\,Z}{f} \qquad (3.60)$$
$$Y = \frac{y^l\,Z}{f} \qquad (3.61)$$
$$\not{Z} = \frac{Z^2\left(u^r - u^l\right)}{f\,l} \qquad (3.62)$$
$$\not{X} = \frac{X\,\not{Z}}{Z} + \frac{u^l\,Z}{f} \qquad (3.63)$$
$$\not{Y} = \frac{Y\not{Z}}{Z} + \frac{v^l Z}{f} \qquad (3.64)$$

3.2.3 Imaging Geometry 3: Slant

In the slant imaging geometry, θ is the degree between the slant and the normal to its line of sight, it is very close to the parallel case, except the assumption $Z^r = Z$ can not be used while calculating the velocity u^s. This problem rises when I try two different methods to recover the correct position of the object, one method is using the perspective projection directly to recover the position and the other is using the same program to recover the object's position when the input data is the correct velocities instead of calculated velocities. The relative error of the two "correct" results is usually less than 1%. But when I

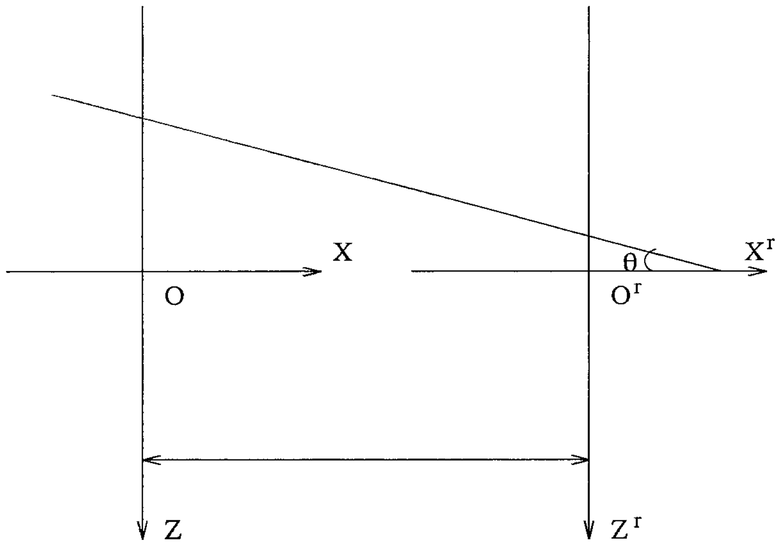


Figure 3.5 Imaging Geometry-3: Slant

try this method in slant imaging geometry with the assumption $Z^r = Z^l = Z$, the relative error of the two ‘‘correct’’ results increase to 5%. To solve this problem, another set of equations is derived here.

The slant imaging geometry can be considered as Figure 3.5. Assuming a point in 3-d world space, its coordinates in the two coordinate systems are (X, Y, Z) and (X^r, Y^r, Z^r) which have the following relation

$$\begin{pmatrix} X^r \\ Y^r \\ Z^r \end{pmatrix} = \left(\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + T \right) \tag{3.65}$$

where T is a translation vector: $T = \begin{pmatrix} -l \\ 0 \\ -l\,tan\,\theta \end{pmatrix}$

The measurement of the variant of the right camera’s position with respect to the left camera’s position δs can be expressed as $\delta s \triangleq \sqrt{(-l)^2} \;=\; |\,l\,|$.

Similarly, following the definition of $u^s = \lim_{\delta s \to 0} \frac{\delta x}{\delta s}$, the below derivitive can be reached

$$\begin{aligned} \frac{u^s}{f} &= \frac{1}{f} \lim_{\delta s \to 0} \frac{\delta x}{\delta s} \\ &\approx \frac{1}{f} \frac{\delta x}{\delta s} \end{aligned}$$

$$
\begin{aligned}
&= \frac{x^r - x^l}{f}\frac{1}{\delta s} \\
&= \left(\frac{X^r}{Z^r} - \frac{X^l}{Z^l}\right)\frac{1}{|\,l\,|} \\
&\quad \left(can\ not\ use\ assumption\ Z^r \approx Z\right) \\
&= \left(\frac{(X-l)}{Z - l\,tan\,\theta} - \frac{X}{Z}\right)\frac{1}{|\,l\,|} \\
&= \frac{-l\,Z + l\,X\,tan\,\theta}{Z\left(Z - l\,tan\,\theta\right)}\frac{1}{|\,l\,|} \\
&= \frac{-l + l\,\frac{x}{f}\,tan\,\theta}{Z - l\,tan\,\theta}\frac{1}{|\,l\,|}
\end{aligned}
$$

So, the expression for Z can be obtained as below

$$Z = l\,tan\,\theta + \left(-1 + \frac{x}{f}\,tan\,\theta\right)\frac{f\,l}{u^s\,|\,l\,|}$$

Similarly, the following six equations can be derived to solve the six unknown motion field quantities in this image geometry

$$Z = l\,tan\,\theta + \left(-1 + \frac{x}{f}\,tan\,\theta\right)\frac{f\,l}{u^s\,|\,l\,|} \tag{3.66}$$

$$X = \frac{x^l\,Z}{f} \tag{3.67}$$

$$Y = \frac{y^l\,Z}{f} \tag{3.68}$$

$$\dot{Z} = \frac{Z^2\left(u^r - u^l\right)}{f\,l} \tag{3.69}$$

$$\dot{X} = \frac{X\,\dot{Z}}{Z} + \frac{u^l\,Z}{f} \tag{3.70}$$

$$\dot{Y} = \frac{Y\dot{Z}}{Z} + \frac{v^l Z}{f} \tag{3.71}$$

3.3 Experimental Techniques and Results

Similarly to determining the optical flow, three experiments are made here to implement recovering motion of objects. These experiments are based on three image geometries discussed above, one is rotation moving, one is parallel moving and the other is slant moving. The settings of experiments are the same as used in determining the optical flow, see Figure 2.3, Figure 2.8, Figure 2.13. Referring to Experiment I, Experiment II, Experiment III made in chapter 2, more detail about settings of experiment, input image sequence and correct optical flow can be obtained in the same way. Also using the information of correct optical flow, correct motion of objects also can be obtained.Error measurement (relative error measurement and angular error measurement) are also used here to analyze the performance of this newly proposed approach. The results are shown in Table 3.1 and Table 3.2 .

Following is the notations used in these Tables. α --- a weight factor between the measurement for the rate of change of image brightness and the measurement for the departure from smoothness in the velocity flow. sw --- means the search window. cw --- means the correlation window. gm --- means the Gaussian mask. deviat --- means the deviation for the statistical characteristic. 1,2,3,4 --- mean the 1st, 2nd, 3rd and 4th image used in these experiments. * --- means the best approach among these three approaches corresponding to one specific experiment setting.

From the results shown in the Table 3.1 and Table 3.2, the following observation can be developed.

- Angular measure of error ψ_e cannot be used here, it is meaningless when there is no way to measure angular deviation just from one direction.

- Rotation geometry algorithm works very well in motion analysis to recovering object position, this can also be proven qualitatively.

  It's known that

$$Z = -\frac{f\, l\, sin\, \theta\, cos\, \theta}{u^s\, sin\, \theta \sqrt{l^2 + \chi^2 \theta^2} + 2f\,(1 - cos\, \theta)}$$

where $u^{*s} = u^s \sqrt{l^2 + \chi^2 \theta^2}/l_x$ is used in this thesis, Z can be expressed in the below

$$\begin{aligned}
Z &= -\frac{f\, l\, sin\,\theta\, cos\,\theta}{u^{*s}\, sin\,\theta\, l_x + 2f\,(1 - cos\,\theta)} \\
\frac{dZ}{du^{*s}} &= -\frac{f\, l\, sin\,\theta\, cos\,\theta\, sin\,\theta\, l_x}{(u^{*s}\, sin\,\theta\, l_x + 2f\,(1 - cos\,\theta))^2} \\
&\quad (Using\ the\ known\ parameters\ in\ this\ setting) \\
&= \frac{12.5(-47.2)\, sin(-2.5)\, cos(-2.5)\, 0.05588\, sin(-2.5)}{(u^{*s}\, 0.05588\, sin(-2.5) + 2 \times 12.5(1 - cos(-2.5)))^2} \\
&\quad (\ u^{*s}\ ranges\ from\ 0\ to\ -0.9444) \\
&\quad (Using\ u^{*s} = -1(pixel/second)\ ) \\
&\approx \frac{0.063067497}{(0.02623146)^2} \\
&\approx 90\ (mm \cdot second/pixel)
\end{aligned}$$

- On the other hand, parallel geometry algorithm works poorly in motion analysis to recover object position, it can also be proven qualitatively.

  It's known that in this geometry, $Z = -\frac{f\,l}{u^s\,|l|}$, similarly, $u^{*s} = u^s\,\sqrt{l^2}/l_x$ is used in this thesis, Z can be expressed as $Z = -\frac{f\,l}{u^{*s}\,l_x}$.

$$\begin{aligned}
\frac{dZ}{du^{*s}} &= \frac{f\,l}{l_x}\frac{1}{(u^{*s})^2} \\
&= \frac{30 \cdot 2.2}{0.05588}\frac{1}{(-1.3895)^2} \\
&\approx 700\ (mm \cdot second/pixel)
\end{aligned}$$

Comparing the above two results, it is shown that parallel imaging geometry algorithm is much more sensitive to the image flow error than rotation imaging geometry algorithm. The parallel imaging geometry algorithm is not suitable to recover the object's position.

Table 3.1 Results of Relative Error (as to position P(X,Y,Z))

		Gradient-based approach Using image 2,3	Correlation-based approach Using image 1,2,3	Correlation-feedback approach Using image 1,2,3
		Iteration = 128 $\alpha = 10$	Iteration = 25 sw = 9 × 9 cw = 5 × 5 gm = 5 × 5	Iteration = 12 Horn-iteration = 30 sw = 5 × 5 cw = 3 × 3 gm = 3 × 3
		ε_e	ε_e	ε_e
Sphere	Z	0.021877 *	0.031524	0.039566
	X	0.022025 *	0.041414	0.051873
	Y	0.022854 *	0.046481	0.053264
Plane	Z	26.399652	212.485062	0.197676 *
	X	7.930244	42.965904	0.166631 *
	Y	9.426798	275.560822	0.176282 *
Slant	Z	2.424380	12.290870	0.387771 *
	X	1.946272	3.682637	0.317726 *
	Y	2.130626	13.863711	0.548148 *

Table 3.2 Results of Angular Error (as to position P(X,Y,Z))

		Gradient-based approach Using image 2,3		Correlation-based approach Using image 1,2,3		Correlation-feedback approach Using image 1,2,3	
		Iteration = 128 $\alpha = 10$		Iteration = 25 sw = 9 × 9 cw = 5 × 5 gm = 5 × 5		Iteration = 12 Horn-iteration = 30 sw = 5 × 5 cw = 3 × 3 gm = 3 × 3	
		ψ_e		ψ_e		ψ_e	
		average	deviat.	average	deviat.	average	deviat.
Sphere	Z	0.00108	0.00052	0.00099	0.00137	0.00145	0.00129
	X	0.04253	0.06494	0.02868	0.05539	0.05608	0.10156
	Y	0.04854	0.06252	0.03594	0.06160	0.04515	0.05112
Plane	Z	1.26144	14.8655	0.01816	0.00121	0.00762	0.00542
	X	3.42467	14.1641	1.59044	1.67008	0.66902	0.83359
	Y	3.88653	14.2059	1.85745	1.88311	0.83339	1.05078
Slant	Z	0.24622	6.35666	9.35175	39.8839	0.01083	0.00876
	X	2.24361	6.80466	9.94329	35.1677	1.04929	1.88105
	Y	2.33847	6.71815	11.1416	38.2559	1.02869	1.29566

CHAPTER 4
CONCLUSION

Through studying three different approaches to determination of optical flow and implementing the associate experiments on real image sequences, some general observations can be obtained as follows.

The gradient-based approach is based on the first order derivatives of the image intensity. Optical flow can be computed by two assumptions: one is intensity-constancy assumption and the other is smoothness-constraint assumption. But it must be noted when implementing this algorithm, the smoothness-constraint assumption is not true on the boundaries in images. The primary difficulty for the gradient-based approach arises from the fact that they are only suitable when the displacements are small compared with the scale of the image intensity variations. However, this requirement is not easily met in real image sequences due to the low temporal sampling rate. That is, because of the limitation of quantization and resolution, the temporal sampling rate cannot be high, otherwise the scale of the image intensity variation would be too small to calculate the optical flow [3]. The gradient-based approach should perform well when boundaries are not too sharp. Experiment I -- Sphere experiment discussed in this thesis proves this observation.

The correlation-based approach is in fact the correlation matching process using sum-of-squared-differences match measurement. The estimate of image velocity can be computed by using the weighted-least-squares estimation. Propagation procedure has to be used to reduce errors, actually the propagation procedure does a much better job of preserving the boundary discontinuities than the gradient-based approach. However, one of the main problems with the SSD-based matching techniques is lacking the ability to estimate displacements which are not the multiple of pixels (one pixel distance is the distance between two consecutive pixels). For image movement involving integers or near integer numbers of displacement, they appear to perform well, but when the motion involves velocities involving a fraction of the pixel, the estimated displacements are often poor.

The correlation-feedback approach, which is based on the correlation-based approach and the idea of feedback, can obtain continuous image by a bilinear interpolation of digital image. It can recover the true image vectors that are not equal to integer pixels, and can make the system robust to noise, and thus raising the estimation accuracy. It can be considered as an improved correlation-based approach. These two correlation approaches can do well for texture picture, and correlation-feedback approach can do better when correct image vectors are not integer number of pixels. Experiment II and Experiment III prove this observation.

It is worth noticing that correct optical flows are needed when the error measurements are used. So far, the unit along the x direction --- the distance between two consecutive pixels along the x direction --- and the unit along the y direction --- the distance between two consecutive pixels along the x direction --- (task of calibration) are not really ``correct'', and the distance between camera and object is still questionable. How to obtain the correct unit along the $x$ and $y$ directions and the distance between the camera and the object is another research subject. That's why that the correct optical flow is ``roughly'' calculated in this thesis.

As to the correlation approaches, the two-step procedure is only based on a pair of images or a sequence of three images. But it is difficult to obtain high accuracy due to noise, digitization error, etc. Many applications require that the estimation algorithms can operate in an on-line and incremental fashion. Therefore in this step, the future approach will be developed based on UOFF and Kalman filter. Kalman filter is a powerful technique for doing incremental, real-time estimation in dynamic systems. It allows the integration of information overtime and it is robust with respect to both system and sensor noises. In the near future, Kalman filter will be used to do motion analysis based on a sequence of many consecutive images.

What have been accomplished in this thesis work can be summarized as follows. In this thesis, three different approaches to determining optical flow and the unified optical flow field method to estimate motion parameters have been studied thoroughly. Two sets of equations for analysis of 3-d motion field based on UOFF, corresponding to parallel imaging geometry and slant imaging geometry, have been derived. Experiments for optical flow determination and motion analysis are conducted on real image sequence input data.

APPENDIX

DERIVATION OF FORMULAE AND C PROGRAMS

The following is the derivation of formulae for calculating correct velocities in rotation geometry.

Figure APPENDIX.1 and Figure APPENDIX.2 show the setting and perspective projection for calculating correct velocity, and the following equations can be established from the above Figures.

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

$$\frac{x}{X} = -\frac{f}{D - Z}$$

$$\frac{y}{Y} = -\frac{f}{D - Z}$$

where $P(X, Y, Z)$ is one point of the object in 3-d world space, $p(x, y)$ is the corresponding point of the image in 2-d image plane, f is the focal length of the CCD camera, D is the distance between the optical center of the camera and the object, a is the long radius of ellipse, b is the short radius of ellipse. After rotating $\Delta\theta$ degree, $P(X, Y, Z)$ becomes $P_2(X_2, Y_2, Z_2)$ and $p(x, y)$ becomes $p_2(x_2, y_2)$. In this problem, $p(x, y)$ is known and $p_2(x_2, y_2)$ is unknown. In the procedure of solving this problem, X, Y, Z,X_2,Y_2 and Z_2 are used as the auxiliary variables.

The procedure of solving this problem can be described as the following

$$b^2X^2 + a^2Y^2 + b^2Z^2 = a^2b^2$$

$$b^2(-\frac{x(D - Z)}{f})^2 + a^2(-\frac{y(D - Z)}{f})^2 + b^2Z^2 = a^2b^2$$

$$b^2x^2(D - Z)^2 + a^2y^2(D - Z)^2 + b^2f^2Z^2 = a^2b^2f^2$$

$$(b^2x^2 + a^2y^2 + b^2f^2)Z^2 - 2(b^2x^2 + a^2y^2)DZ + (b^2x^2 + a^2y^2)D^2 - a^2b^2f^2 = 0$$

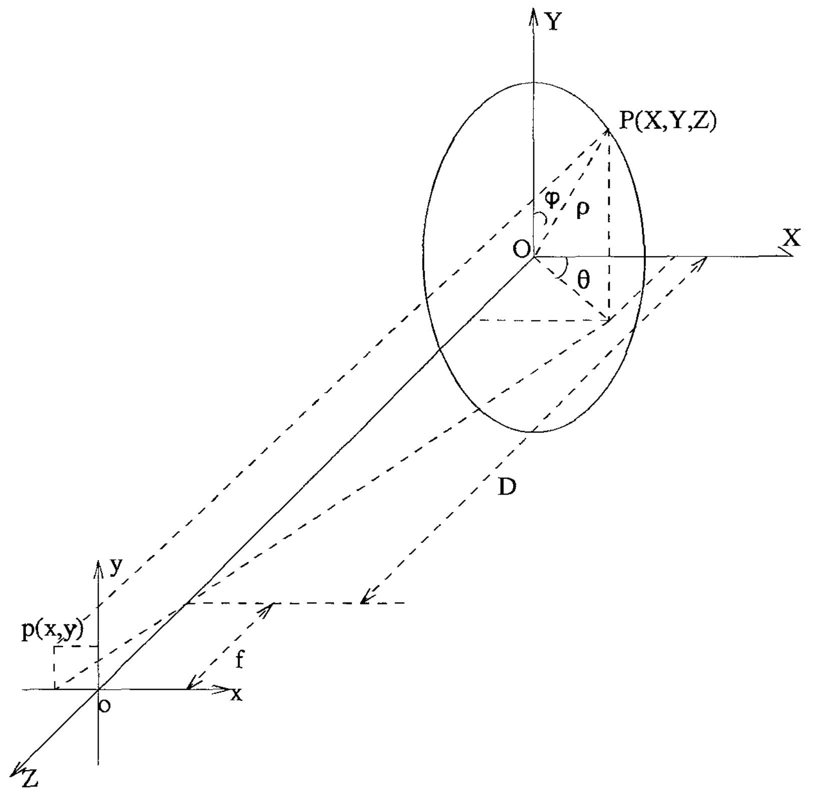


Figure APPENDIX.1 Setting for Calculating Correct Velocity

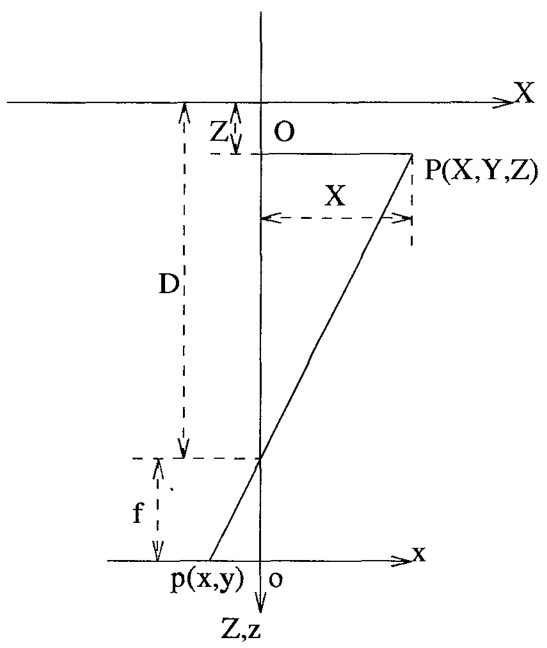


Figure APPENDIX.2 Perspective Projection for Calculating Correct Velocity

For the sake of simplicity, let

$$\begin{aligned} A &= b^2x^2 + a^2y^2 + b^2f^2 \\ B &= -2(b^2x^2 + a^2y^2)D \\ C &= (b^2x^2 + a^2y^2)D^2 - a^2b^2f^2 \end{aligned}$$

So, the auxiliary variable Z can be calculated as the following

$$Z = \frac{-B + \sqrt{B^2 - 4AC}}{2A}$$

It is worth to notice that $Z = \frac{-B-\sqrt{B^2-4AC}}{2A}$ can be ignored since just one point which is near the camera should be used.

So far, given one point of image in the image plane, the corresponding point in 3-d world space can be calculated as the following

$$\begin{aligned} Z &= \frac{-B + \sqrt{B^2 - 4AC}}{2A} \\ X &= -\frac{x(D-Z)}{f} \\ Y &= -\frac{y(D-Z)}{f} \end{aligned}$$

Rotating $\Delta\theta$ degree, the relation between $P(X, Y, Z)$ and $P_2(X_2, Y_2, Z_2)$ can be described as

$$\begin{pmatrix} X_2 \\ Y_2 \\ Z_2 \end{pmatrix} = \begin{pmatrix} cos\Delta\theta & 0 & -sin\Delta\theta \\ 0 & 1 & 0 \\ sin\Delta\theta & 0 & cos\Delta\theta \end{pmatrix} \times \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

The relation can be derived as the following.

First, it can be assumed that

$$\begin{aligned} X &= \rho\, sin\varphi\, cos\theta \\ Y &= \rho\, cos\varphi \\ Z &= \rho\, sin\varphi\, sin\theta \end{aligned}$$

Then, if the ellipsoid rotates count-clockwise, it can be gotten

$$\begin{aligned}
X_2 &= \rho\, sin\varphi\, cos(\theta - \Delta\theta) \\
&= \rho\, sin\varphi\, (cos\theta\, cos\Delta\theta + sin\theta\, sin\Delta\theta) \\
&= \rho\, sin\varphi\, cos\theta\, cos\Delta\theta + \rho\, sin\varphi\, sin\theta\, sin\Delta\theta \\
&= X\, cos\Delta\theta + Z\, sin\Delta\theta \\
Y_2 &= \rho\, cos\varphi \\
&= Y \\
Z_2 &= \rho\, sin\varphi\, sin(\theta - \Delta\theta) \\
&= \rho\, sin\varphi (sin\theta\, cos\Delta\theta - cos\theta\, sin\Delta\theta) \\
&= \rho\, sin\varphi\, sin\theta\, cos\Delta\theta - \rho\, sin\varphi\, cos\theta\, sin\Delta\theta \\
&= -X\, sin\Delta\theta + Z\, cos\Delta\theta
\end{aligned}$$

From now on, the $p_2(x_2, y_2)$ point can be computed as

$$\begin{aligned}
x_2 &= -\frac{fX_2}{D - Z_2} \\
y_2 &= -\frac{fY_2}{D - Z_2}
\end{aligned}$$

Assuming the time interval between one frame and the next frame is one second, then the correct optical flow $\vec{U_c}(u_c, v_c)$ can be calculated as $u_c = \Delta\, x = x_2 - x\ (pixel/second)$, $v_c = \Delta\, y = y_2 - y\ (pixel/second)$.

The following is the C program for calculation of correct velocities in rotation geometry.

```
/*
 * To calculate the correct motion of an ellipse image.
 */
#include <stdio.h>
#include <math.h>
#include <rasterfile.h>

/* The unit length per pixel in the image plane */
```

```
#define UNIT_X (0.01397 * 4.0)
#define UNIT_Y (0.0116416 * 4.0)

int size_x,size_y;  /* the size of the image  */

float D,f;  /* the parameters of the geometry of the camera */

float theta;    /* the rotate angle of the ellipse */

float a,b,aa,bb;  /* parameter of the ellipse in image plane */

typedef struct
{
  struct
    {
      float x,y;
    } point;
  struct
    {
      float X,Y,Z;
    } Point;
} mapping;

typedef struct
{
  float delta_x;
  float delta_y;
} velocity;

/*
 * Map a point in the image plane to the real world.
 */
void MapFrom2Dto3D (map)
mapping *map;
{
  float Z;
  float tmp = b * b * map -> point.x * map ->point.x
            + a * a * map -> point.y * map -> point.y;
  float A = tmp + b * b * f * f ;
  float B = -2 * D * tmp;
  float C = tmp * D * D - a * a * b * b * f * f;

  /* here, we solve the classic quadratic equation
     A * Z ^2 + B * Z + C = 0; */

  float delta = B * B - 4 * A * C;
  if ( delta < 0 )
    panic("No real solution in this problem. \n");
  map -> Point.Z = Z = (- B + sqrt ((double)delta))*0.5 / A;
  map -> Point.X = - map -> point.x * ( D - Z) / f;
  map -> Point.Y = - map -> point.y * ( D - Z) / f;
}

/*
 * The motion on the image plane at (x,y), see the comments
 * of MapFrom2Dto3D
 */
void motion (x,y,vp)
float x,y;
velocity *vp;
{
  mapping map;
  float X1,Y1,Z1, X2,Y2,Z2;
  map.point.x = x;
  map.point.y = y;

  MapFrom2Dto3D ( &map );
  X1 = map.Point.X;
  Y1 = map.Point.Y;
  Z1 = map.Point.Z;

  X2 = X1 * cos (theta) - Z1 * sin (theta);
  Z2 = X1 * sin (theta) + Z1 * cos (theta);
  Y2 = Y1;

  vp -> delta_x = -f * X2 / (D - Z2) - x;
  vp -> delta_y = -f * Y2 / (D - Z2) - y;
}

int on_the_ellipse (x,y)
float x,y;
{
  if ( ( x * x /  + y * y / bb ) <= 1.0 )
    return 1;
  return 0;
}

main(int argc, char **argv)
{
  struct rasterfile head;
  char *on_it;
  int fd1,fd;
  int i,j;    /* the integer i, j coordinates of the image,*/
              /* (0,0) is the upper-left corner.*/
  velocity v;
  float *vels,x,y;
```

```
if ( argc != 9 )
  panic("Usage:a.out size_x size_y a b D f theta output.\n");
size_x = atoi (argv[1]);
size_y = atoi (argv[2]);
D = (float)atof (argv[5]);
f = (float)atof (argv[6]);
theta = atof (argv[7]) * M_PI / 180.0;
a = (float)atof (argv[3]) * UNIT_X *D / f;
b = (float)atof (argv[4]) * UNIT_Y *D / f;
aa = f * f * a * a / ( D * D );
bb = f * f * b * b / ( D * D );

if ( (vels = (float *)malloc(size_x*size_y*sizeof(velocity)
                           + sizeof(float) * 6) ) == NULL)
  panic("Not enough memory.\n");

head.ras_magic = RAS_MAGIC;
head.ras_width = size_x;
head.ras_height = size_y;
head.ras_depth = 8;
head.ras_length = size_x*size_y;
head.ras_type = RT_STANDARD;
head.ras_maptype = RMT_NONE;
head.ras_maplength = 0;
if ( (on_it = (char *)malloc(size_x * size_y) ) == NULL)
  panic("Not enough memory.\n");
if ( (fd1 = creat("on_it" , 0644) ) == -1)
  panic("Wrong when create on_it file.\n");
write (fd1 , &head, sizeof(head) );

vels[0] = vels [2] = size_x;
vels[1] = vels [3] = size_y;
vels[4] = vels [0] - vels [2];
vels[5] = vels [1] - vels [3];

for ( j = 0; j < size_y; j++ )
  for ( i = 0; i < size_x ; i++ )
    {
      x = ( i - size_x / 2.0 ) * UNIT_X ;
      y = ( -j + size_y / 2.0 ) * UNIT_Y;
      if (on_the_ellipse(x,y) )
        {
          motion(x, y, &v);
          vels[ j * size_x * 2 + i * 2 + 6]
                                  = v.delta_x / UNIT_X;
          vels[ j * size_x * 2 + i * 2 + 6 + 1]
                                  = v.delta_y / UNIT_Y;
```

```
              on_it[ j * size_x + i] = 0xff;
            }
          else
            {
              vels[ j * size_x * 2 + i * 2 + 6 ] = 0.0;
              vels[ j * size_x * 2 + i * 2 + 6 + 1 ] = 0.0;
              on_it[ j * size_x + i ] = 0;
            }
        }

    if ( (fd = creat (argv[8] , 0644) ) == -1 )
      panic("Wrong when create the on_it file.\n");
    if ( write (fd, (char *)vels,size_x*size_y*sizeof(velocity)
                                  + sizeof (float)*6 )
        != size_x*size_y*sizeof(velocity) + sizeof(float)*6 )
      panic("Wrong when write the on_it file.\n");

    write (fd1, on_it, size_x * size_y);

    printf("You Got the Ellipse.\n");
    exit (0);
}

panic (error_message)
char *error_message;
{
  puts(error_message);
  exit (-1);
}
```

The following is the C program for recovery of object position in 3-d space and correspond velocities in these three geometries.

```
/*
 * To calculate the exact position (X,Y,Z) of the object in
 * 3-D space and correspond velocities.
*/

#include <stdio.h>
#include <math.h>

#define UNIT_X (0.01397 * 4.0)
#define UNIT_Y (0.0116416 * 4.0)
#define UNIT_L 1.0
```

```
int size_x, size_y;    /* the size of image in pixel number.*/

float focus;           /*the parameter of the camera. */

float length;          /*the distance between the right and */
                       /*left cameras.*/
int factor;            /* Experiment I--1; Experiment II--2 */
                       /* Experiment III--3 */

main(int argc, char **argv)
{
  float us,vs, ur,vr, ul,vl, X,Y,Z, Z_vel,X_vel,Y_vel,theta;
  float theta_sin,theta_cos,theta_tan;

  FILE *ifp[6];
  FILE *ofp[6];
  char outname[256];

  int i,j;
  float x,y;

  if ( argc != 14)
    no_luck("You lose, Usage:position size_x size_y focus \
  length theta factor us vs ur vr ul vl output_base\n");

  size_x = atoi(argv[1]);
  size_y = atoi(argv[2]);
  focus  = atof(argv[3]);
  length = atof(argv[4]);
  theta  = atof(argv[5]);
  factor = atoi(argv[6]);

  theta = (theta) * M_PI / 180.0 ;
  theta_sin = (float) sin(theta) ;
  theta_cos = (float) cos(theta) ;
  theta_tan = (float) cos(theta) ;

  for ( i = 0; i < 6; i++)
    {
      if ( (ifp[i] = fopen(argv[i+7], "r")) == NULL)
        {
          perror(argv[i+7]);
          exit (-1);
        }
      sprintf(outname, "%s.%d", argv[12], i);
      if ( (ofp[i] = fopen(outname, "w")) == NULL)
        {
          perror(outname);
        exit(-1);
      }
  }

for ( j=0; j<size_y;j++)
  for ( i=0; i<size_y;i++)
    {
      x = ( i - size_x/2 )*UNIT_X;
      y = ( -j + size_y/2 )*UNIT_Y;

      if ( fread ( (char *)&us, sizeof (us),1,ifp[0]) != 1 )
        no_luck("You lose when read.\n");

      if ( fread ( (char *)&vs, sizeof (us),1,ifp[1]) != 1 )
        no_luck("You lose when read.\n");

      if ( fread ( (char *)&ur, sizeof (us),1,ifp[2]) != 1 )
        no_luck("You lose when read.\n");

      if ( fread ( (char *)&vr, sizeof (us),1,ifp[3]) != 1 )
        no_luck("You lose when read.\n");

      if ( fread ( (char *)&ul, sizeof (us),1,ifp[4]) != 1 )
        no_luck("You lose when read.\n");

      if ( fread ( (char *)&vl, sizeof (us),1,ifp[5]) != 1 )
        no_luck("You lose when read.\n");

      if (factor = 1)
        Z = - focus*length*theta_sin*theta_cos / (us*UNIT_L
             *UNIT_X*theta_sin + 2*focus*(1 - theta_cos ) );
      if (factor = 2)
        Z = -focus * length / ( us * UNIT_X);
      if (factor = 3)
        Z = length*theta_tan + length*(x*theta_tan/focus- 1)
          * focus/(us*UNIT_L*UNIT_X) ;

      X = x * Z / focus;
      Y = y * Z / focus;

      if (factor = 1)
        Z_ = (-ul*theta_cos + ur)*UNIT_X*Z*Z/(focus*length);
      if (factor =2)
        Z_vel = (ur - ul) * UNIT_X *Z * Z / (focus*length);
      if (factor = 3)
        Z_vel = (ur - ul) * UNIT_X *Z * Z / (focus*length);

      X_vel = X*Z_vel/Z + ul*UNIT_X*Z/focus;
```

```
            Y_vel = Y*Z_vel/Z + vl*UNIT_Y*Z/focus;
          }


    if ( fwrite( (char *)&Z, sizeof(Z) , 1 , ofp[0]) != 1)
      no_luck("You lose when write.\n");

    if ( fwrite( (char *)&X, sizeof(Z) , 1 , ofp[1]) != 1)
      no_luck("You lose when write.\n");

    if ( fwrite( (char *)&Y, sizeof(Z) , 1 , ofp[2]) != 1)
      no_luck("You lose when write.\n");

    if ( fwrite( (char *)&Z_vel, sizeof(Z) , 1 , ofp[3]) != 1)
      no_luck("You lose when write.\n");

    if ( fwrite( (char *)&X_vel, sizeof(Z) , 1 , ofp[4]) != 1)
      no_luck("You lose when write.\n");

    if ( fwrite( (char *)&Y_vel, sizeof(Z) , 1 , ofp[5]) != 1)
      no_luck("You lose when write.\n");

    return 0;
  }

  no_luck(char *msg)
  {
    puts(msg);
    exit(-1);
  }
```

REFERENCES

1. Horn, B. K. P., and B. G. Schunck. "Determining Optical Flow." *Artificial Intelligence* 14 (1981): 185-203.

2. Singh, A. "An Estimation-theoretic Framework For Image-flow Computation." *Proceeding of the 3rd International Conference on Computer Vision,* Osaka, Japan. (1990, Dec. 4-6).

3. Pan, J. N., Y. Q. Shi, and C. Q. Shu. "A New Approach of Computing Optical Flow Based on Analysis of Errors of the Correlation-based Method." *Technical Report No. 19, Electronic Imaging Laboratory,* Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, New Jersey. (1992).

4. Barron, J. L., D. J. Fleet, and B. B. Beauchemin. "Performance of Optical Flow Techniques." *Technical Report No. 299,* Department of Computer Science, University of Western Ontario, London, Ontario and Department of Computer Science, Queens University, Kingston, Ontario. (1992).

5. Shu, C. Q., and Y. Q. Shi. "A New Approach to Motion Analysis From a Sequence of Stereo Images." *Technical Report No. 18, Electronic Imaging Laboratory,* Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, New Jersey. (1990).

6. Shu, C. Q., and Y. Q. Shi. "Computation of Motion From Stereo Image Sequence Using the Unified Optical Flow Field." *SPIE's 1990 International Symposium on Optical and Optoelectronic Applied Science and Engineering,* San Diego, CA. (1990).

7. Shu, C. Q., and Y. Q. Shi. "On unified Optical Flow Field." *Pattern Recognition,* Vol. 24, No. 6 (1991, June): 579-586.