STATISTICAL METHODS OF MOTION ESTIMATION FROM OMNI-DIRECTIONAL IMAGE SEQUENCES

by

Feng Chen

B. S., Zhongshan University, 1990

Submitted in partial fulfillment of the requirements
for the Degree of Master of Science in the Department
of Computer Science at Brown University

Providence, Rhode Island
May 2001

Abstract

Recent research has proposed estimating structure from motion using omni-directional image sequences which facilitates accurate estimation. This project investigated the geometry involved in non-linear projections, implemented various algorithms to perform coordinate/optical flow projection and 3-D motion parameter estimation, and proposed a direct method to compute motion on the omni-directional images.

This project by Feng Chen is accepted in its present form by the Department of Computer Science as satisfying the research requirement for the degree of Master of Science.

Date ______________ ____________________________________

Michael Black, Director

Acknowledgements

I would like to thank my advisor Dr. Michael Black for his constant guidance and support, and Dr. Seth Teller of Massachusetts Institute of Technology for a helpful discussion and providing camera information and omni-directional image sequences.

Chapter 1

Introduction

1.1 Background

The problem of determining world structure and motion from a sequences of images is an active field in computer vision. Information about the structure and the motion is encoded into the image sequences. The interpretation of such information consists of forming object hypothesis and recovering the motion parameters. The results can be used in many applications such as 3-D image reconstruction [10] and robot navigation [11].

The common approach to computing motion from image sequences includes two phases. The first phase is to compute the optical flow field (velocity vectors) from the image sequences. The second phase is to analyze the optical flow field and determine the motion parameters.

The interpretation of the optical flow is the main focus of this project. Most approaches that have been previously investigated involve estimations on perspectively projected images. Researches done by J. Gluckman and S. K. Nayar [8] proposed estimating motion parameters on a spherical surface. This project verifies the spherical approach and also proposes direct estimation on the omni-directional image and spherical surface.

1.2 Previous Works

1.2.1 The Brightness Constancy Assumption

The brightness constancy assumption is to assume that the brightness of a small surface patch (esp. a pixel) is not changed by motion. It is a fundamental constraint in motion estimation for data conservation. Let $I(x, y, t)$ be the image intensity or brightness at point (x, y) at time t. Expanding the derivative of the brightness I with respect to t leads to

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{1.1}$$

Let (u, v) denote the horizontal and vertical instantaneous velocity of point p, $\frac{dx}{dt}$ and $\frac{dy}{dt}$, and

I_x, I_y, I_t denote the partial derivatives of the brightness function with respect to x, y and t. The brightness constancy assumption can be written as

$$I_x u + I_y v + I_t = 0 \tag{1.2}$$

(I_x, I_y) forms the local brightness gradient vector.

The data conservation error can be formulated as the sum-of-square differences over all regions (points) in the image. Let E be the error function, it leads the following data conservation constraint:

$$E(u,v) = \sum (I_x u + I_y v + I_t)^2 \tag{1.3}$$

where the summation is over all pixels in the image. This is usually called "Gradient-Based Method".

Another way to derive brightness constancy constraint is the so-called "Correlation Method." The brightness constancy assumption can be expressed as:

$$I(x,y,t) = I(x + u\delta t, y + v\delta t, t + \delta t) \tag{1.4}$$

assuming image velocity is approximately constant in the small neighborhood. The data conservation error function is

$$E(u,v) = \sum (I(x,y,t) - I(x + u\delta t, y + v\delta t, t + \delta t))^2 \tag{1.5}$$

Note that the above equation becomes 1.3 when $I(x + u\delta t, y + v\delta t, t + \delta t)$ is approximated by Taylor expansion and dropped the higher order terms.

As a special case, consider affine flow. Let

$$u = a_0 + a_1 x + a_2 y \tag{1.6}$$

$$v = a_3 + a_4 x + a_5 y \tag{1.7}$$

be the model of image flow of a region or the entire image. Regression methods can be easily applied to the parametric form of the brightness constancy constraint and find parameters $a_0, a_1, ..., a_5$ that minimize the error function.

However, the brightness constancy assumption is often violated in the real world, primarily due to the existence of multiple motions. There are many proposals to work out this problem, as shown in the sub-sections below.

1.2.2 Motion Under Planar Perspective Projection

Consider the motion of a world point in the Cartesian coordinate system. Let $P = (X, Y, Z)$ denote the coordinates of this point at time t, and $P' = (X', Y'Z')$ the coordinates at time t'. The motion of this point can be modeled as first rotating about the three axes and then translating along the three directions:

$$P' = RP + T \tag{1.8}$$

where

$$R = \begin{bmatrix} 1 & -\mathit{\Omega}_z & \mathit{\Omega}_y \\ \mathit{\Omega}_z & 1 & -\mathit{\Omega}_x \\ -\mathit{\Omega}_y & \mathit{\Omega}_x & 1 \end{bmatrix} \tag{1.9}$$

is the approximated rotation matrix when the rotation is small. And

$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{1.10}$$

is the translation vector along three directions.

Bruss and Horn [4] developed a method to estimate motion using images through a perspective camera. Consider the projection of a world point $P = (X, Y, Z)$ to a image plane at the distance of 1 in Figure 1.1 , the coordinate of the image point is $p = (x, y, 1)$. The perspective projection in the Cartesian coordinate system can be described as:

$$x = \frac{X}{Z} \tag{1.11}$$

$$y = \frac{Y}{Z} \tag{1.12}$$

Let $p = (x, y)$ and $p' = (x', y')$ denote the projection of P and P', it is easy to compute the location of p' given the rotation matrix and translation vector:

$$x' = \frac{X'}{Z'} = \frac{x - \Omega_z y + \Omega_y + \frac{T_x}{Z}}{-\Omega_y x + \Omega_x y + 1 + \frac{T_z}{Z}} \tag{1.13}$$

$$y' = \frac{Y'}{Z'} = \frac{y + \Omega_z x - \Omega_x + \frac{T_y}{Z}}{-\Omega_y x + \Omega_x y + 1 + \frac{T_z}{Z}} \tag{1.14}$$

Let Δx and Δy denote the displacements on x and y directions:

$$\Delta x = x' - x = \frac{-\Omega_x xy + \Omega_y(1 + x^2) - \Omega_z y + \frac{T_x - T_z x}{Z}}{-\Omega_y x + \Omega_x y + 1 + \frac{T_z}{Z}} \tag{1.15}$$

$$\Delta y = y' - y = \frac{-\Omega_x(1 + y^2) + \Omega_y xy + \Omega_z z + \frac{T_y - T_z y}{Z}}{-\Omega_y x + \Omega_x y + 1 + \frac{T_z}{Z}} \tag{1.16}$$

If the translation on z direction Tz is far smaller than the depth of the point Z, the field of view is not very large, and the rotation is small, the above equations can be approximated as:

$$\Delta x = -\Omega_x xy + \Omega_y(1 + x^2) - \Omega_z y + \frac{T_x - T_z x}{Z} \tag{1.17}$$

$$\Delta y = -\Omega_x(1 + y^2) + \Omega_y xy + \Omega_z z + \frac{T_y - T_z y}{Z} \tag{1.18}$$

Now let $\tau = (\tau_x, \tau_y, \tau_z)$ and $\omega = (\omega_x, \omega_y, \omega_z)$ denote the instantaneous velocity of translations and rotations of a rigid object in the scene related to the camera, i.e., $\tau \leftarrow dT/dt$, $\omega \leftarrow d\Omega/dt$, then

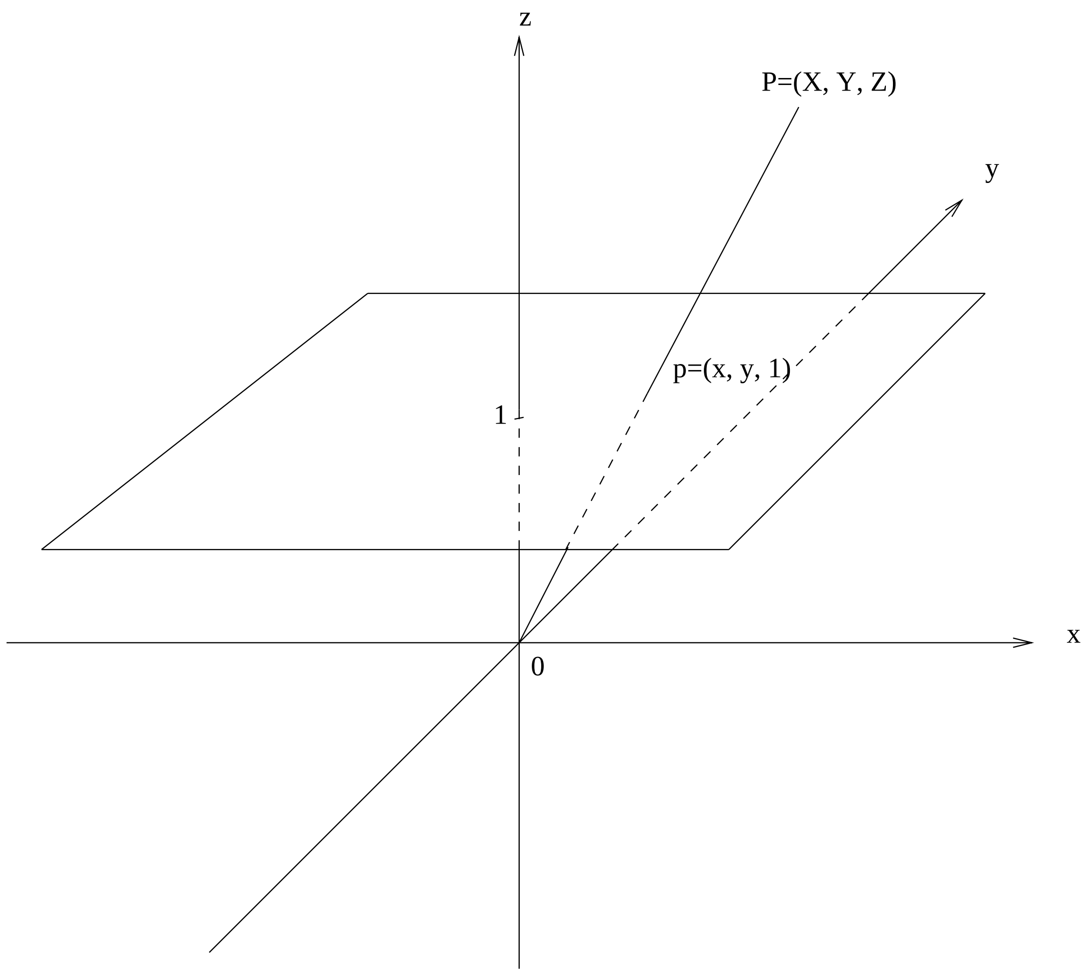

Figure 1.1: View-centered coordinate systems. The world point $P = (X, Y, Z)$ is projected onto the image plane as point $p = (x, y, 1)$

the velocity of the projected point $p = (x, y)$ on the image is exactly:

$$\frac{dx}{dt} = -\omega_x xy + \omega_y(1 + x^2) - \omega_z y + \frac{\tau_x - \tau_z x}{Z} \tag{1.19}$$

$$\frac{dy}{dt} = -\omega_x(1 + y^2) + \omega_y xy + \omega_z z + \frac{\tau_y - \tau_z y}{Z} \tag{1.20}$$

One can observe that the velocity is a linear function of translational and rotational parameters τ and ω. Since the displacement forms of T and W are not used in this project, in the rest of this report T and W will be used to denote τ and ω.

1.2.3 Robust Estimation and Multiple Motions

It is easy to see that in the above formulations of the brightness conservation constraint, a single motion is assumed. However, in the real world multiple motions usually exist in a single frame.

While it is possible to shrink the field of view so that only one motion is included, a smaller region is more susceptive to noise. This is often referred as "the Generalized Aperture Problem:" [9]

1. The field of view must be large enough to sufficiently constrain the solution;
2. The field of view must be small enough to avoid the existence of multiple motions.

In addition, the spatial coherence constraint also suffers from the same problem. It assumes that the flow changes gradually within a small neighborhood. However, multiple motions in the same frame create boundary discontinuity, which violates the smoothness assumption of the spatial coherence constraint.

There are many approaches to dealing with such conflict. In [5], Adiv used an approach that consisted of two stages. In the first stage the flow field is partitioned into connected segments of flow vectors, where each segment is consistent with a rigid motion of a roughly planar surface. In the second stage, segments are grouped under the hypothesis that they are introduced by a single, rigid moving object. The hypothesis is tested by searching for 3-D motion parameters that are compatible with all the segments in the corresponding group.

Jepson and Black [2] assumed a layered representation of and modeled the constraint lines within a region as a mixture of distributions corresponding to the different layers. Then a modified version of the EM-algorithm is used to compute a maximum likelihood estimate for the various motion parameters.

In [9], Black and Anandan developed a framework based on robust estimation that addresses violations of the brightness constancy and spatial smoothness assumptions caused by multiple motions and applied this framework to a number of common optical flow formulations: area-based regression, correlation, and regularization with motion discontinuities.

1.2.4 Motion Computation from Optical Flow

Most motion estimation methods comprise two phases: the computation of optical flow and the extraction of motion parameters. Under perspective projection, the motion equations are

$$u = \frac{dx}{dt} = -\Omega_x xy + \Omega_y(1+x^2) - \Omega_z y + \frac{T_x - T_z x}{Z} \tag{1.21}$$

$$v = \frac{dy}{dt} = -\Omega_x(1+y^2) + \Omega_y xy + \Omega_z z + \frac{T_y - T_z y}{Z} \tag{1.22}$$

where U is the 2×1 vector to denote optical flow of a point on the image plane, T and Ω are 3×1 motion parameter vectors, p is the 3×1 vector of the image point ($z = 1$) and Z the depth of the point in the world.

By using vectorial notation, it can be re-written as

$$U = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -x_2 \end{bmatrix} (\frac{T}{Z} + \Omega \times p) \tag{1.23}$$

Bruss and Horn [4] derived a depth independent constraint to remove depth Z from the above equation and obtained a bilinear constraint on T and Ω on every pixel. Later MacLean and Jepson [17] derived the same bilinear constraint by applying different manipulation:

$$T^T(p \times U) + (T \times p)^T(p \times \Omega) = 0 \tag{1.24}$$

From the above equation a least squares estimate of rotation can be obtained as a function of translation. Substituting this rotation estimate back into the bilinear constraint gives a non-linear constraint on translation. The translation can be estimated by minimizing the non-linear constraint over all velocity vectors. Since both depth and translational parameter are unknown, the translational parameter T can only be recovered up to a scale. Without losing generality, constraint $|T| = 1$ is used in the estimation algorithm.

Jepson and Heeger [1] developed another linear algorithm which is called linear subspace method. Given optical flow sampled at N discrete points in the image, one can construct a set of constraint vectors τ_i

$$\tau_i = \sum_{k=1}^{N} c_{ik}(U_k \times p_k) \tag{1.25}$$

such that τ_i and T are orthogonal to each other:

$$\tau_i \cdot T = 0 \tag{1.26}$$

The choice of $c_i = [c_{i1}c_{i2}...c_{ik}]^T$ should suffice that they are orthogonal to all quadratic polynomials of x_k and y_k. For N discrete points in the image, there are $N - 6$ linear constraints. The estimate of T is the eigenvector corresponding to the smallest eigenvalue of $\sum \tau_i \tau_i^T$.

Soatto and Brockett [15] investigated the structure from motion problem under the condition that large noise is present. Similar to Bruss and Horn's work, they developed a bilinear projection method that iteratively estimates rotation and translation. The difference from Bruss and Horn's method is that they used a spherical projection model. The outline of this algorithm can be found in Chapter 4 of this report.

Tian et. al. [16] compared several popular approaches for egomotion computation and obtained quantitative benchmarks on their performance.

1.2.5 Motion Estimation under Non-Linear Projection

One problem in motion estimation from optical flow is the sensitivity to noise in the second step. Despite the tradeoffs as indicated in the Generalized Aperture Problem, a large field of view always facilitates motion estimation. With omni-directional image sequences, the motion field contains global patterns that do not always manifest themselves in a small field of view, such as the focus points of expansion and contraction. By using global patterns, the estimation tends to be steadier and more accurate.

Another problem with planar perspective projection is the ambiguity of translational and rotational fields. As illustrated in Figure 1.2, translation parallel to the image plane and rotation about the vertical axis produce similar motion fields when the field of view is small. They are virtually indistinguishable in the presence of noise. However, with large field of view such as a spherical surface they appear to be quite distinct.

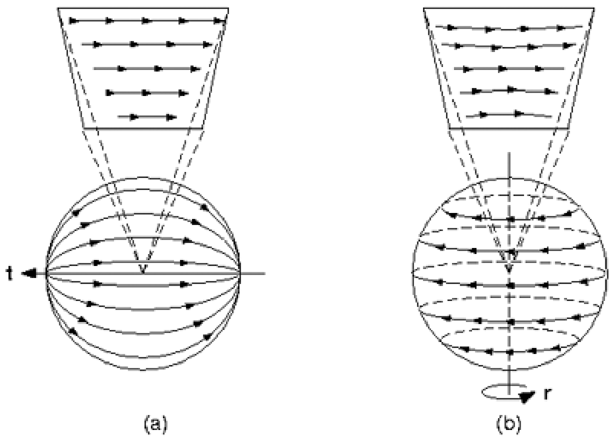

Figure 1.2: Ambiguity of flow field. (a) is translation flow being projected onto a sphere and a plane parallel to the direction of translation. (b) is rotation flow being projected onto a sphere and a plane parallel to the direction of rotation. (Figure from [8])

For this reason several methods have been proposed to compute motion using large field of view or under non-linear projections. Yen and Huang [6] mapped 2-D image onto a unit sphere and derived a non-iterative solution based on geometry and algebraic manipulation. Yagi et. al. used a hyperbolic omni-directional camera to compute motion under the assumption that the camera moves in a horizontal plane. Gluckman and Nayar [8] developed a method to map the image velocity vectors to a sphere and then estimate apply the existing egomotion algorithm on the spherical surface.

1.3 Objective of the Research

The objective of this thesis research is to understand various methodologies and algorithms in the area of motion estimation, study and investigate motion estimation under non-linear projection models, formulate and experiment with new methods that could contribute to this subject.

Chapter 2

The Projection Models

2.1 Omni-Directional Cameras

It is well know in computer vision that increasing the field of view enhances the capability of many vision applications such as motion estimation and 3-D image reconstruction. It has been further proved that uncertainty in motion estimates depends on the shape of the image plane. A spherical image plane provides lower ambiguity than a planar image plane as shown in Figure 1.2.

Other applications in which conventional cameras do not fit very well include robotic navigation, video surveillance, immersive telepresence, video conferencing, mosaicing, and map building. For example, in urban scene survey [10], a long baseline is usually assumed. With a conventional camera it may require many more images to be captured than that with an omni-directional camera. Another example is autonomous vehicle navigation, where an omni-direction camera is used to capture a wide range of traffic situation and compute relative motions.

Omni-directional vision can be realized with rotating imaging systems, dioptric systems or catadioptric systems. Rotating imaging systems revolve traditional cameras about the camera pinhole and the stitch the images together to produce a panoramic view [7] [3]. It is comparable to a common technology in photography, where photographers rotates camera on a tripod and shoots consecutive frames. In the motion estimation domain, however, this method does help because it essentially captures static scenes.

Dioptric systems use fish-eye lens to increase the field of view. Fish-eye lens have very short focal length and when used with conventional camera, can capture a field of view that is approximately a hemisphere. However, one intrinsic problem of fish-eye lens is the hardness to design one with a single center of projection. Although some of them have small viewpoint locus, the acquired image does not permit the construction of a totally distortion-free perspective image of the scene.

Catadioptric systems are combinations of mirrors and lens. They use a reflecting surface to enhance the field of view and perspective lens to capture the image. A number of methods have been proposed to build practical sensing systems. Nalwa [12] proposed aligning four planar mirrors in the shape of a pyramid and achieving a single center of projection; Nayar [13] implemented a parabolic based omni-directional imaging system with parabolic mirror, an orthographic lens system and a CCD camera.

2.2 Parabolic Projection

Figure 2.1 illustrates the optical system of a parabolic omni-directional camera. The focal point of the parabola lies on the origin of the coordinate system. Axis z is the axis of the parabola. Consider a world point $P = (X, Y, Z)$ heading to the focal point of the parabolic mirror, intersecting the mirror surface at point $\hat{p} = (\hat{x}, \hat{y}, \hat{z})$, then being reflected and falling onto the image plane at point $p = (x, y)$. The geometric properties of a parabola cause a ray pointing to the focal point to be reflected parallel to the axis of the parabola. Let θ denote the polar angle between the incoming ray and the z axis and ϕ the azimuth angel. The relation between θ and the z coordinate of the reflecting point $\hat{p}$ is

$$\tan\theta = \frac{\rho}{\hat{z}} \tag{2.1}$$

where

$$\rho = \sqrt{\hat{x}^2 + \hat{y}^2} \tag{2.2}$$

is the distance between the coordinate origin and point $\hat{p}$. Coordinate $\hat{z}$ can be solved [14] easily through the differential equation that describes the parabolic surface:

$$\hat{z} = \frac{h^2 - \rho^2}{2h} \tag{2.3}$$

It would be convenient to express $\hat{p}$ in spherical coordinate. One important property is that the spherical coordinate separates the depth of a world point from other two coordinates, which would be shown useful later in motion estimation.

$$\theta = \arccos\frac{\hat{z}}{\sqrt{\hat{x}^2 + \hat{y}^2 + \hat{z}^2}} \tag{2.4}$$

$$\phi = \arctan\frac{\hat{y}}{\hat{x}} \tag{2.5}$$

$$\rho = \sqrt{\hat{x}^2 + \hat{y}^2} = \frac{h}{1 + \cos\theta} \tag{2.6}$$

The inverse of the above equations can be solved and expressions for $\hat{x}, \hat{y}$ obtained as:

$$\hat{x} = \frac{h\sin\theta\cos\phi}{1 + \cos\theta} \tag{2.7}$$

$$\hat{y} = \frac{h\sin\theta\sin\phi}{1 + \cos\theta} \tag{2.8}$$

Point $p = (x, y)$ on the image plane has the same coordinate values as point $\hat{p}$, assuming an orthographic projection from the parabola to the image plane:

$$x = \hat{x} \tag{2.9}$$

$$y = \hat{y} \tag{2.10}$$

Given the above geometry one can figure out the projection of a 3-D object to the omni-directional image plane. Consider a point in the world $P = (X, Y, Z)$, which can be written in spherical coordinates:

$$\rho_P = \sqrt{X^2 + Y^2 + Z^2} \tag{2.11}$$

$$\theta_P = \arctan \frac{\sqrt{X^2 + Y^2}}{Z} \tag{2.12}$$

$$\phi_P = \arctan \frac{Y}{X} \tag{2.13}$$

Point P can be mapped onto a point P_0 on the omni-directional image plane:

$$P_0 = (x_0, y_0) \tag{2.14}$$

where

$$x_0 = \frac{h \sin \theta_P \cos \phi_P}{1 + \cos \theta_P} \tag{2.15}$$

$$y_0 = \frac{h \sin \theta_P \sin \phi_P}{1 + \cos \theta_P} \tag{2.16}$$

2.3 Spherical Projection

The projection of a world point $P = (X, Y, Z)$ onto a unit sphere can be best described using spherical coordinates. Assuming the center of the unit sphere is also the origin of the Cartesian coordinates, equations for converting between Cartesian and spherical coordinates are

$$\rho = \sqrt{X^2 + Y^2 + Z^2} \tag{2.17}$$

$$\theta = \arctan \frac{\sqrt{X^2 + Y^2}}{Z} \tag{2.18}$$

$$\phi = \arctan \frac{Y}{X} \tag{2.19}$$

and

$$X = \rho \sin \theta \cos \phi \tag{2.20}$$

$$Y = \rho \sin \theta \sin \phi \tag{2.21}$$

$$Z = \rho \cos \theta \tag{2.22}$$

Let $\hat{P} = (\hat{\rho}, \hat{\theta}, \hat{\phi})$ be the projected point on the sphere, the relations between this point and the world point P are

$$\hat{\rho} = 1 \tag{2.23}$$

$$\hat{\theta} = \theta \tag{2.24}$$

$$\hat{\phi} = \phi \tag{2.25}$$

and

$$\begin{aligned}
\hat{X} &= \frac{X}{\sqrt{X^2+Y^2+Z^2}} && (2.26)\\
\hat{Y} &= \frac{Y}{\sqrt{X^2+Y^2+Z^2}} && (2.27)\\
\hat{Z} &= \frac{Z}{\sqrt{X^2+Y^2+Z^2}} && (2.28)\\
& && (2.29)
\end{aligned}$$

The Cartesian relation is usually written as the simple form:

$$\hat{P} = \frac{P}{\|P\|} \tag{2.30}$$

where $\|P\|$ is the depth of the world point.

The geometry of spherical projection is illustrated in Figure 2.2.

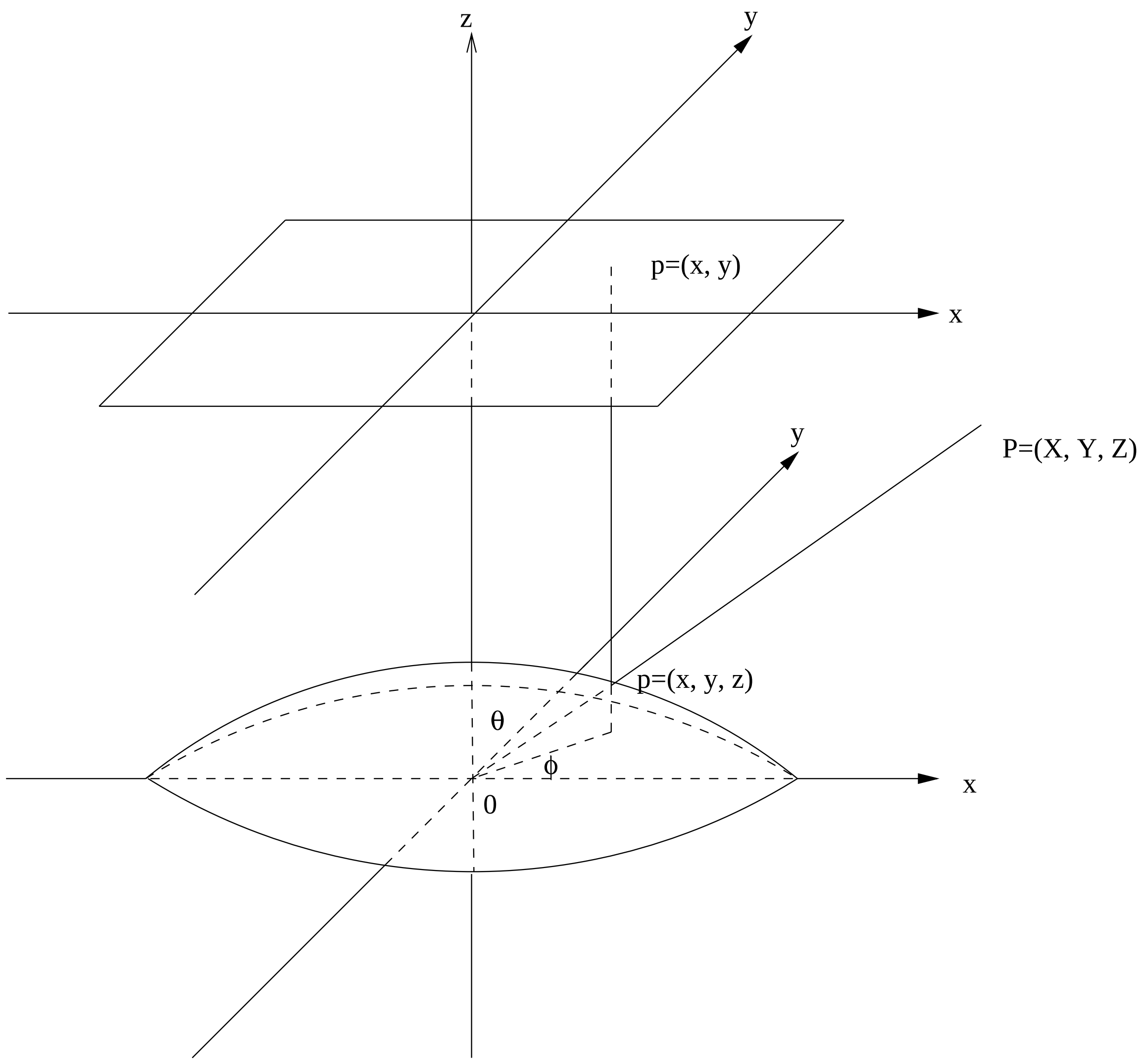

Figure 2.1: Parabolic projection. The world point $P = (X, Y, Z)$ is first projected onto the parabolic surface at point $\hat{p} = (\hat{x}, \hat{y}, \hat{z})$, then reflected image plane as point $p = (x, y)$

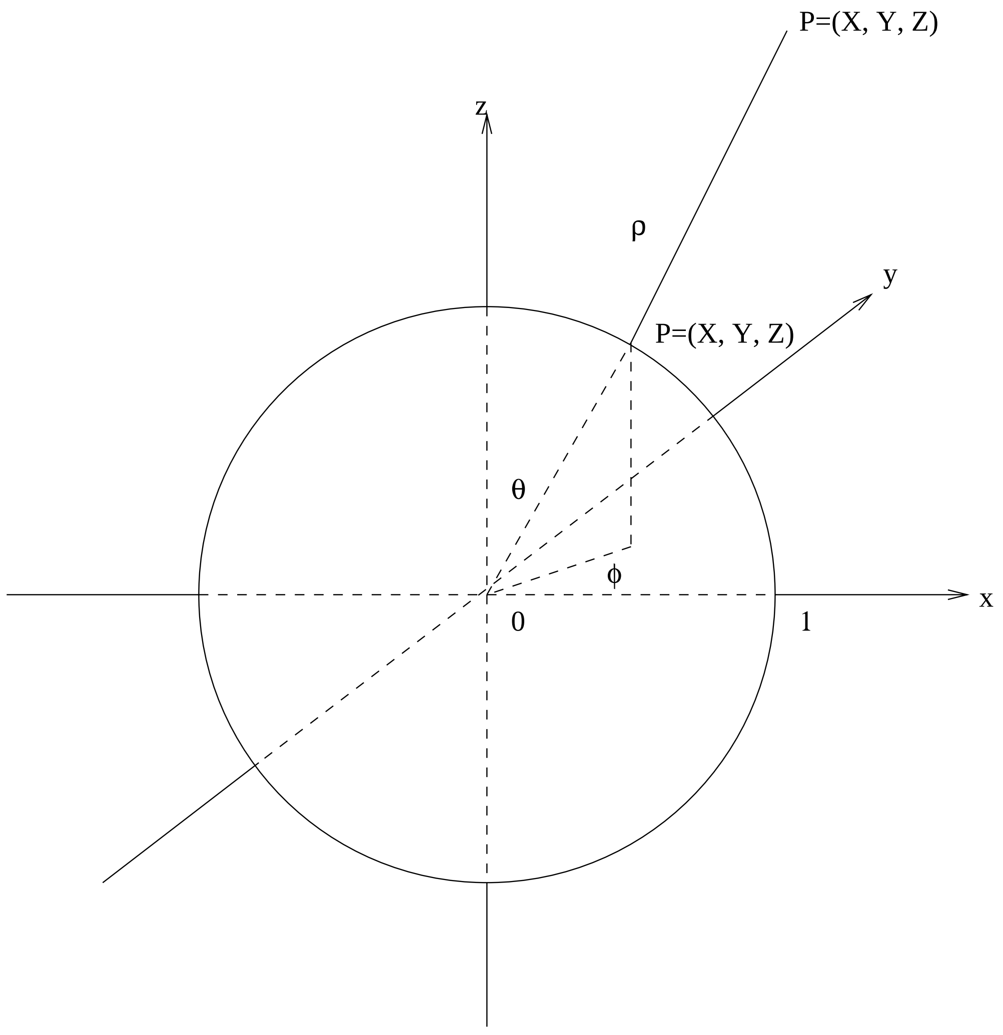

Figure 2.2: Spherical projection. The world point $P = (X, Y, Z)$ is projected onto the spherical surface at point $\hat{P} = (\hat{X}, \hat{Y}, \hat{Z})$.

Chapter 3

Motion Estimation with Omni-Directional Images

There are three approaches to computing motion parameters from a sequence of omni-directional images:

1. Projecting images onto a unit sphere, computing optical flow, and estimating motion parameters.
2. Computing optical flow on the omni-directional images, projecting the images as well as the optical flow onto the unit sphere, and estimating motion parameters.
3. Formulating the error function for the brightness constancy constraint, estimating motion parameters that minimize this error function. This is called the direct method.

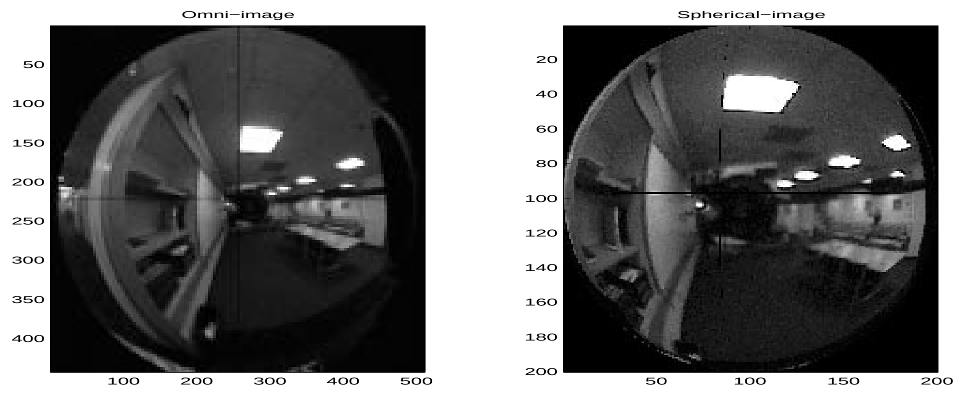

Figure 3.1: The omni-directional image and its projection on the spherical surface.

Figure 3.1 shows one frame of the original omni-directional image and the projected spherical image. Figure 3.2 is the original and mapped optical flow in both horizontal and vertical directions.

The mapped flow also includes z direction for depth change velocity because it lies on a 3-D sphere.

3.1 Estimation by Projecting the Optical Flow onto the Sphere

In section 1.2.5 the advantages of using omni-directional image and computing motion under non-linear projection have been discussed. One such approach is to compute optical flow on the omni-directional image plane and then map it to a unit sphere. Motion estimation is performed on the surface of this unit sphere. There are several reasons to do this:

1. One problem to compute motion parameter is that T is coupled with scene depth $||P||$. By using epipolar constraint, the depth of the scene point can be removed from the motion equation [8].
2. As indicated in 1.2.5, on the spherical surface the ambiguity between translational and rotational motion can be better distinguished.
3. Existing ego-motion algorithms can be easily adapted to spherical projection.

Motion estimation on the sphere has two divergences as indicated in the beginning of this chapter. In [8], the author believed that projecting the image onto the spherical surface introduced artifacts. Note that most of the point on the sphere do not map to exact pixels on the omni-directional image plane. Instead, they fall into some places in the middle. Interpolation is usually necessary to obtain relatively accurate intensity values. Even though, the mapping is considered as some kind of approximation. In approach 1 of the above discussion, the optical flow will be computed on this approximated mapping. On the other hand, in approach 2 the projection of motion field does not have such problem simply because only those vectors existed on the omni-directional image plane will be mapped and used to compute motion parameters.

Figure 3.3 shows the geometry of the projection. Recall in Chapter 1 of this report, the instantaneous velocity of a 3-D world point $P = (X, Y, Z)$ is described as rotations about three axes and translations along the three directions:

$$\frac{dP}{dt} = -T - \Omega \times P \tag{3.1}$$

where Ω and T are motion parameters. Chapter 2 shows that the spherical perspective projection of P is

$$\hat{P} = \frac{P}{\|P\|} \tag{3.2}$$

where $\hat{P} = (\hat{X}, \hat{Y}, \hat{Z})$ is the coordinates of the projected point on the sphere and $\|P\| = \sqrt{X^2 + Y^2 + Z^2}$ is the depth of the scene point. Taking derivatives of 3.2 with respect to t and substituting in 3.1 leads to the following motion equation [8]:

$$U(\hat{P}) = \frac{1}{\|P\|}((T \cdot \hat{P})\hat{P} - T) - \Omega \times \hat{P} \tag{3.3}$$

where $U(\hat{P})$ is the velocity of point $\hat{P}$ on the spherical surface.

In the above equation, $\hat{P}$ and $U(\hat{P})$ can be obtained by projecting the coordinates and velocity of the corresponding point on the omni-directional image onto the unit sphere. Once $\hat{P}$ and $U(\hat{P})$ are known, T, Ω and depth $\|P\|$ can be estimated using some optimization methods. Let $p = (x, y)$ denote the coordinates of the omni-directional image point (2-D point for short), and (u, v) be the associated velocity. The polar angel θ and azimuth angel ϕ can be determined as follows:

$$\theta = 2\arctan\frac{\sqrt{x^2+y^2}}{h} \tag{3.4}$$

$$\phi = \arctan\frac{y}{x} \tag{3.5}$$

Since the unit sphere has a radius of 1, the coordinates of the corresponding point on the sphere are:

$$\hat{X} = \sin\theta\cos\phi \tag{3.6}$$

$$\hat{Y} = \sin\theta\sin\phi \tag{3.7}$$

$$\hat{Z} = \cos\theta \tag{3.8}$$

Mapping the velocity vector consists two sub-steps. First, the coordinate system is changed to spherical:

$$\frac{d\hat{X}}{dt} = \frac{\partial\hat{X}}{\partial\theta}\frac{d\theta}{dt} + \frac{\partial\hat{X}}{\partial\phi}\frac{d\phi}{dt} = \cos\theta\cos\phi\frac{d\theta}{dt} - \sin\theta\sin\phi\frac{d\phi}{dt} \tag{3.9}$$

$$\frac{d\hat{Y}}{dt} = \frac{\partial\hat{Y}}{\partial\theta}\frac{d\theta}{dt} + \frac{\partial\hat{Y}}{\partial\phi}\frac{d\phi}{dt} = \cos\theta\sin\phi\frac{d\theta}{dt} + \sin\theta\cos\phi\frac{d\phi}{dt} \tag{3.10}$$

$$\frac{d\hat{Z}}{dt} = \frac{\partial\hat{Z}}{\partial\theta}\frac{d\theta}{dt} + \frac{\partial\hat{Z}}{\partial\phi}\frac{d\phi}{dt} = -\sin\theta\frac{d\theta}{dt} \tag{3.11}$$

Second, the spherical coordinate system is changed to the (x, y) coordinates on the image plane:

$$\frac{d\theta}{dt} = \frac{\partial\theta}{\partial x}\frac{dx}{dt} + \frac{\partial\theta}{\partial y}\frac{dy}{dt} = \frac{2hx}{\sqrt{x^2+y^2}(h^2+x^2+y^2)}\frac{dx}{dt} + \frac{2hy}{\sqrt{x^2+y^2}(h^2+x^2+y^2)}\frac{dy}{dt} \tag{3.12}$$

$$\frac{d\phi}{dt} = \frac{\partial\phi}{\partial x}\frac{dx}{dt} + \frac{\partial\phi}{\partial y}\frac{dy}{dt} = -\frac{y}{x^2+y^2}\frac{dx}{dt} + \frac{x}{x^2+y^2}\frac{dy}{dt} \tag{3.13}$$

It is easy to use a Jacobian matrix to simplify the transformation equations. The Jacobian relates partial derivatives in one coordinate system to those in another. Let

$$J_1 = \begin{bmatrix} \frac{2hx}{\sqrt{y^2+x^2}(h^2+y^2+x^2)} & \frac{2hy}{\sqrt{y^2+x^2}(h^2+y^2+x^2)} \\ -\frac{y}{y^2+x^2} & \frac{x}{y^2+x^2} \end{bmatrix} \tag{3.14}$$

and

$$J_2 = \begin{bmatrix} \cos\theta\cos\phi & -\sin\theta\sin\phi \\ \cos\theta\sin\phi & \sin\theta\cos\phi \\ -\sin\theta & 0 \end{bmatrix} \tag{3.15}$$

The mapping of the velocity vector can be written as:

$$U(\hat{P}) = J_2 J_1 [u\ v]^T \tag{3.16}$$

With the knowledge of $\hat{P}$ and $U(\hat{P})$, the ego-motion problem is to estimate Ω and T such that they minimize some error function. An algorithm that outlines the above procedure is as follows.

1. For each image point p=(x,y)
   (a) Compute θ, ϕ for the ray
   (b) Compute $\hat{X}$, $\hat{Y}$ and $\hat{Z}$
   (c) Compute Jacobian matrices J1, J2
   (d) Compute velocity vector U on the sphere
2. Invoke estimation method to determine the unknown parameters T and Ω from $\hat{P}$ and $U(\hat{P})$.

3.2 Estimation by Projecting the Image onto the Sphere

In the above section the optical flow is computed on the omni-directional image and then mapped to a spherical projection model using the Jacobian of the transformation. Alternatively, the image itself can be projected onto the sphere and then compute the optical flow. Although mapping the image may introduce artifacts, in some cases they may not be severe. Experiment result shows the motion parameter estimates are very close to those computed by the method in 3.1.

3.3 Direct Estimation on the Omni-Directional Images

The most straightforward way to motion estimation is to compute the parameters directly from the omni-directional image sequence without computing the optical flow. The general idea is to formulate the error function of the brightness constancy constraint (Equation 1.3). This error function is a function of motion parameters T, Ω and scene depth R at each point, plus known variables such as image coordinates and derivatives of image intensity. T, Ω and R are the best estimates that minimize that error function.

Consider a point (x, y) on the omni-directional image plane and its associated velocity vector (u, v). Through coordination transformation, x, y can be expressed as

$$x = \frac{h \sin\theta \cos\phi}{1 + \cos\theta} \tag{3.17}$$

$$y = \frac{h \sin\theta \sin\phi}{1 + \cos\theta} \tag{3.18}$$

and the velocity vector becomes

$$u = \frac{dx}{dt} = \frac{\partial x}{\partial \theta}\frac{d\theta}{dt} + \frac{\partial x}{\partial \phi}\frac{d\phi}{dt} + \frac{\partial x}{\partial \rho}\frac{d\rho}{dt} \tag{3.19}$$

$$v = \frac{dy}{dt} = \frac{\partial y}{\partial \theta}\frac{d\theta}{dt} + \frac{\partial y}{\partial \phi}\frac{d\phi}{dt} + \frac{\partial y}{\partial \rho}\frac{d\rho}{dt} \tag{3.20}$$

Since x, y are not related to depth ρ, the above equations are just:

$$u = \frac{h(\cos\phi\frac{d\theta}{dt} - \sin\theta\sin\phi\frac{d\phi}{dt})}{1+\cos\theta} \tag{3.21}$$

$$v = \frac{h(\sin\phi\frac{d\theta}{dt} + \sin\theta\cos\phi\frac{d\phi}{dt})}{1+\cos\theta} \tag{3.22}$$

On the other hand, the instantaneous velocity of a 3-D world point $P = (X, Y, Z)$ is

$$\frac{dP}{dt} = -T - \Omega \times P \tag{3.23}$$

where Ω and T are motion parameters. Re-write it in component form:

$$\frac{dX}{dt} = -T_x - \Omega_y Z + \Omega_z Y \tag{3.24}$$

$$\frac{dY}{dt} = -T_y - \Omega_z X + \Omega_x Z \tag{3.25}$$

$$\frac{dZ}{dt} = -T_z - \Omega_x Y + \Omega_y X \tag{3.26}$$

Now change it to spherical coordinates:

$$X = R\sin\theta\cos\phi \tag{3.27}$$

$$Y = R\sin\theta\sin\phi \tag{3.28}$$

$$Z = R\cos\theta \tag{3.29}$$

where R, θ, ϕ are indicated as in previous Figure. The left-hand side of the above equations are

$$\frac{dX}{dt} = R\cos\theta\cos\phi\frac{d\theta}{dt} - R\sin\theta\sin\phi\frac{d\phi}{dt} + \sin\theta\cos\phi\frac{dR}{dt} \tag{3.30}$$

$$\frac{dY}{dt} = R\cos\theta\sin\phi\frac{d\theta}{dt} + R\sin\theta\cos\phi\frac{d\phi}{dt} + \sin\theta\sin\phi\frac{dR}{dt} \tag{3.31}$$

$$\frac{dZ}{dt} = -R\sin\theta\frac{d\theta}{dt} + \cos\theta\frac{dR}{dt} \tag{3.32}$$

and the right-hand side becomes:

$$\frac{dX}{dt} = -Tx - Oy\,R\cos\theta + Oz\,R\sin\theta\sin\phi \tag{3.33}$$

$$\frac{dY}{dt} = -Ty - Oz\,R\sin\theta\cos\phi + Ox\,R\cos\theta \tag{3.34}$$

$$\frac{dZ}{dt} = -Tz - Ox\,R\sin\theta\sin\phi + Oy\,R\sin\theta\cos\phi \tag{3.35}$$

Therefore, $d\theta/dt$ and $d\phi/dt$ can be solved as follows:

$$\frac{d\theta}{dt} = \frac{\cos\theta\sin\phi Oy\,R - \cos\phi Ty - Oz\,R\sin\theta + \cos\phi Ox\,R\cos\theta + Tx\,\sin\phi}{R\sin\theta} \tag{3.36}$$

$$\frac{d\phi}{dt} = -\frac{-Tz\,\sin\theta + \cos\theta Ty\,\sin\phi + \cos\theta\cos\phi Tx - Ox\,R\sin\phi + Oy\,R\cos\phi}{R} \tag{3.37}$$

Substituting this back to the equation 3.19 and 3.20 to solve dx/dt and dy/dt:

$$\begin{aligned}\frac{dx}{dt} &= -\frac{h}{R(1+\cos\theta)}[(1+\cos\theta\cos^2\phi-\cos^2\phi)Tx \\ &+(\cos\theta\cos\phi\sin\phi-\cos\phi\sin\phi)Ty \\ &-\sin\theta\cos\phi Tz+R(\cos\theta\cos\phi\sin\phi-\cos\phi\sin\phi)\Omega_x \\ &+R(\cos^2\phi+\cos\theta-\cos\theta\cos^2\phi)\Omega_y-R\sin\theta\sin\phi\Omega_z] \qquad (3.38)\\ \frac{dy}{dt} &= -\frac{h}{R(1+\cos\theta)}[(\cos\theta\cos\phi\sin\phi-\cos\phi\sin\phi)T_x \\ &+(\cos\theta-\cos\theta\cos^2\phi+\cos^2\phi)T_y \\ &-\sin\theta\sin\phi T_z+R(\cos^2\phi-\cos\theta\cos^2\phi-1)\Omega_x \\ &+R(\cos\phi\sin\phi-\cos\theta\cos\phi\sin\phi)\Omega_y+R\sin\theta\cos\phi\Omega_z] \qquad (3.39)\end{aligned}$$

where θ, ϕ can be written in terms of x and y:

$$\theta = \arccos\frac{h^2-\hat{x}^2-\hat{y}^2}{h^2+\hat{x}^2+\hat{y}^2} \qquad (3.40)$$

$$\phi = \arctan\frac{\hat{y}}{\hat{x}} \qquad (3.41)$$

The estimate of T, Ω and R are the set of values that minimize the following error function:

$$E=\sum(I_x\frac{dx}{dt}+I_y\frac{dy}{dt}+I_t)^2 \qquad (3.42)$$

where the summation is over all pixels in the image. This function can be minimized by using non-linear optimization methods.

An alternative way is to formulate the above function directly under spherical coordinates:

$$E=\sum(I_\theta\frac{d\theta}{dt}+I_\phi\frac{d\phi}{dt}+I_t)^2 \qquad (3.43)$$

where $\frac{d\theta}{dt}$ and $\frac{d\phi}{dt}$ are from equation 3.36 and 3.37.

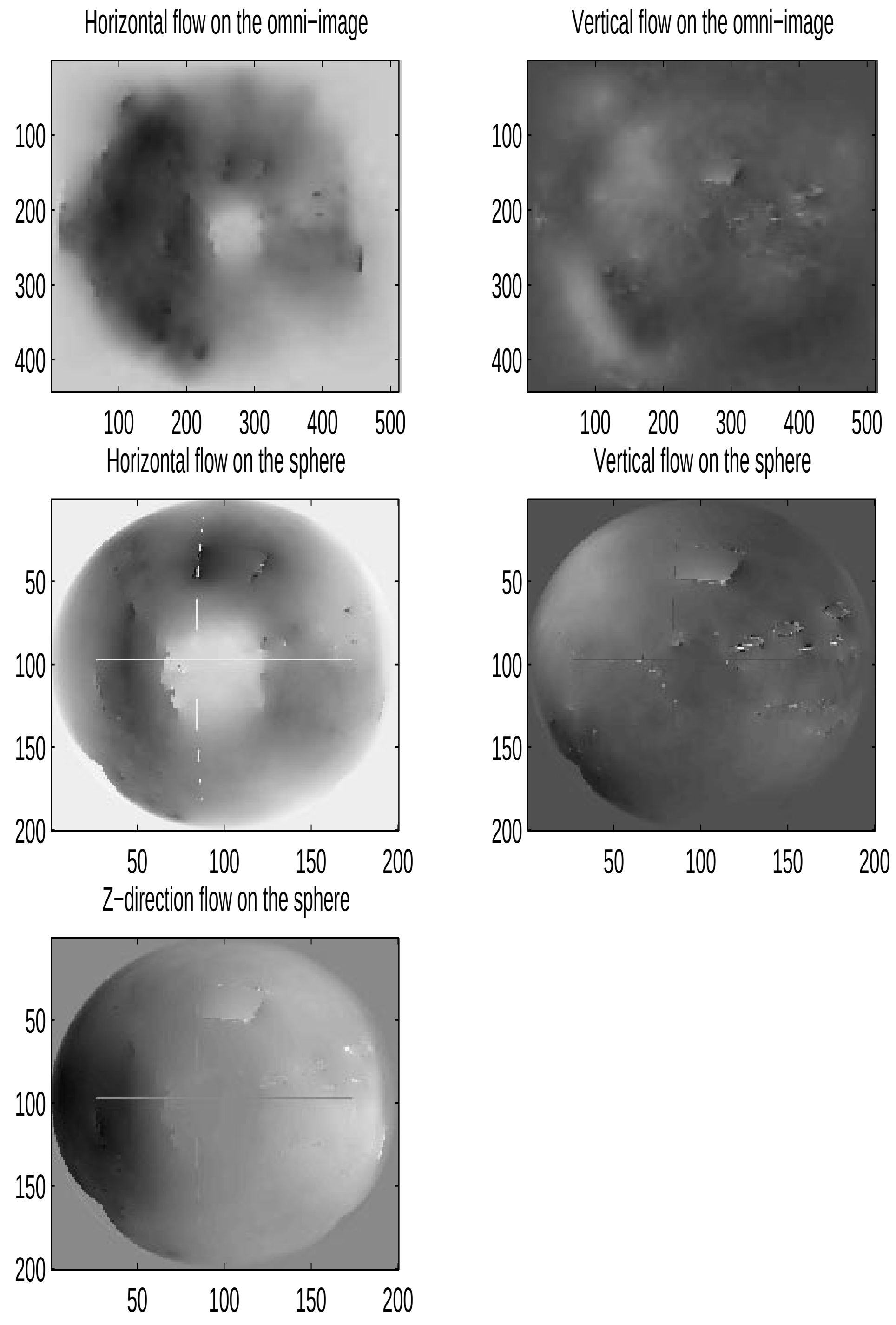

Figure 3.2: Optical flow on the omni-directional image plane and the spherical surface.

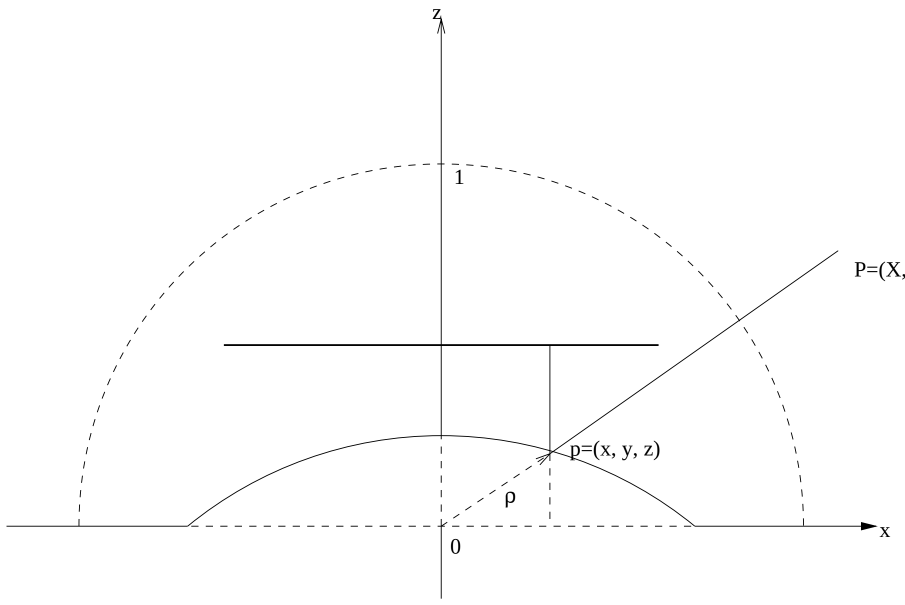

Figure 3.3: Computing optical flow on the sphere.

Chapter 4

Experiments

4.1 Projection Geometries

Experiment data were from MIT Laboratory of Computer Science. The camera uses a perspective lens system to capture the rays reflected by the parabolic surface. Transformation between a device point (x_d, y_d) (measured in physical units) and an image point (x, y) is:

$$x = f_x x_d + c_x \tag{4.1}$$

$$y = f_y y_d + c_y \tag{4.2}$$

where f_x and f_y are focal length of the perspective imaging system. They are different because the pixels are not square. c_x and c_y are the principal point in pixels.

The following procedure outlined the transformation of a point on the omni-directional image to a point on the parabolic surface, i.e., the intersection of the ray and the parabolic mirror.

1. Take a pixel (x, y) in the image, first transform its coordinates to the physical measurement using the transformation (4.1) and (4.2).

2. Offset (x_d, y_d) by the mirror center x_{mirror}, y_{mirror} and produce the x and y coordinates for the point on the parabolic surface:

$$\hat{x} = x_d - x_{mirror} \tag{4.3}$$

$$\hat{y} = y_d - y_{mirror} \tag{4.4}$$

3. Solve the z coordinate for the point on the parabolic surface:

$$\hat{z} = \frac{h^2 - \hat{x}^2 - \hat{y}^2}{2h} \tag{4.5}$$

A series of experiments have been carried out to understand and verify the geometry of the camera model. One such experiment is to take a rectangular image as input, project it to the omni-directional image plane using the camera geometry, then re-project the omni image back to the rectangular plane. Figure 4.1 shows the result of the three images. Artifacts can be observed in the reconstructed image, especially in the region that is far from the center of the image. This is because the mapping between the rectangular image plane and the omni-directional image plane is not one-to-one.

Due to this fact the algorithm should iterate over the points on the target plane, compute the corresponding original pixel and take the intensity of that pixel directly or through interpolation. A simple version of linear interpolation was used in the experiments. If the target point has a (fractional) coordinate of (i, j), then the intensity value I_{ij} can be computed from the four points that are immediately adjacent to this point:

$$I_{ij} = (I_{i_0 j_0}(i_1 - i) + I_{i_1 j_0}(i - i_0))(j_1 - j) + (I_{i_0 j_1}(i_1 - i) + I_{i_1 j_1}(i - i_0))(j - j_0) \tag{4.6}$$

where $I_{i_0 j_0}$, $I_{i_1 j_0}$, $I_{i_0 j_1}$ and $I_{i_1 j_1}$ are the intensity values of the four adjacent points of (i, j).

4.2 Computing Motion on the Unit Sphere by Projecting Optical Flow

In this experiment two omni-directional images are used as input. First, the dense optical flow is computed by the Robust Flow [9] program. Then the image coordinates and the optical flow are projected onto the unit sphere using the derivations outlined in Chapter 3. Finally, the coordinates and velocity on the sphere are used to estimate translation and rotation by the bilinear constraint algorithm.

The first sequence (Figure 4.2) is consisted with two images of an office scene. They have been taken by rotating the omni-directional camera horizontally. Theoretically, the only motion parameter that is not zero is the Ω_y term, which corresponds to the rotation about the y-axis. The recovered translation and rotation are:

$$T = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.7}$$

$$\Omega = \begin{bmatrix} 0.0000911 \\ -0.0008408 \\ -0.0001685 \end{bmatrix} \tag{4.8}$$

One way to verify the motion estimate is to reconstruct the optical flow and use it to warp the first image towards the second. The warping error can be calculated and compared to the warping error with the original optical flow. In this sequence, the reconstructed flow has an RMS warping error of 7.1113, comparing to the RMS warping error of the original image flow, 7.6814. The smaller error in the reconstructed flow is primarily due to the noise.

An interesting observation is that the RMS error for the original image pair is only 6.2457. However, the rotational motion is obvious.

In another sequence (Figure 4.2), both rotational and translational motions exist. The estimated motion by the bilinear constraint algorithm is as follows:

$$T = \begin{bmatrix} -0.9907 \\ 0.1339 \\ 0.0230 \end{bmatrix} \tag{4.9}$$

$$\Omega = \begin{bmatrix} 0.0001 \\ 0.0021 \\ -0.0032 \end{bmatrix} \tag{4.10}$$

One can observe a large horizontal motion in the image sequence which is corresponding to the T_x term. Motions on the y and z directions and rotations are too small to be observed by human eyes. The warping error using the reconstructed flow is 8.9386, comparing to the warping error of the original optical flow 8.6749. As comparison, the error between two (unwarped) images is 11.0242.

However, the recovered optical flow is not accurate because the local extremum corresponding to the so-called "bas-relief ambiguity" is dominant. In this case, the depth of the scene cannot be fully recovered. The normalized depth is the best estimate about the shape. It is primarily due to the existence of noise in the original frames and the optical flow.

4.3 Computing Motion on the Unit Sphere by Projecting Image Only

Another experiment uses the same sequence which has both translation and rotation. Two omni-directional images are projected onto the unit sphere. Then the optical flow is computed using the projected image sequence. Same bilinear algorithm has been used to estimate motion. The recovered parameters are:

$$T = \begin{bmatrix} -0.9919 \\ 0.1099 \\ 0.0630 \end{bmatrix} \tag{4.11}$$

$$\Omega = \begin{bmatrix} 0.0001 \\ 0.0019 \\ 0.0002 \end{bmatrix} \tag{4.12}$$

They are very close to the estimation result of the above section. In addition, the recovered optical flow, though still not accurate for the same reason, produces a more stable warped image than the one in the above section. The result somehow contradicts that of Gluckman and Nayar's prediction about mapping the image to the sphere and then computing optical flow. One possible reason might be the numeric error introduced during the projection of the velocity field in method 3.1, especially when θ, ϕ close to zero or $\pi/2$.

4.4 Bilinear Constraint Algorithm

Two versions of motion estimation algorithms have been implemented. The first one is Jepson and Heeger's linear subspace algorithm. However, the estimation by this algorithm seems to have larger error. For example, it gives a large motion along the z axis, which cannot be observed from the image sequence intuitively. The reasons why this algorithm fails might be (a) this algorithm was developed based on planar perspective projection model; (b) it only uses horizontal and vertical optical flow, while on the sphere there are three velocities instead of two.

Another version of the estimation algorithm is adopted from Soatto and Brockett's [15] bilinear constraint algorithm. Let x_i denote the coordinate of a point $\hat{P}_i$ on the sphere and v_i the associated velocity vector. Construct operator $\hat{x}_i$ such that $x_i \times a = \hat{x}_i a$ where a is any vector. Let $y_i = -\hat{x}_i v_i$ and $\lambda_i = 1/\|X_i\|$ where $\|X_i\|$ is the depth of the corresponding world point of $\hat{P}_i$. The outline of the algorithm can be described as follows:

1. Check for data consistency. The velocity vector should be orthogonal to the direction vector of the point, i.e., the velocity vector lies on the tangent plane of the point vector. If error is greater than some threshold, stop.

2. Check for pure rotation. Assuming zero translation and estimate the rotation using least-square method. If error is smaller than some threshold, stop and use the estimated rotation.

3. Initialize $T = [0\ 0\ 0]'$ and $\Omega = [0\ 0\ 0]'$.

4. Choose a threshold for convergence test and iteratively estimate T and Ω:

$$\begin{aligned} T_k &= argmin \sum \|x_i T'(y_i - \hat{x}_i^2 \Omega_k)\|^2 && (4.13) \\ \Omega_{k+1} &= (\sum \hat{x}_i^2 T_k T_k' \hat{x}_i^2)^{-1} (\sum \hat{x}_i^2 T_k T_k' y_i) && (4.14) \\ k &= k+1 && (4.15) \end{aligned}$$

5. Check for local extrema. Compute the residuals for the three eigenvectors for $\sum (y_i - \hat{x}_i^2 \Omega_k)(y_i - \hat{x}_i^2 \Omega_k)'$ and choose the one that generates the smallest residual as T.

6. Check for bas-relief ambiguity. If all $\lambda_i > 0$, stop. Otherwise reduce the minimal value from all λ_i so that all $\lambda_i > 0$. Use this to estimate the best T and Ω and go to the iteration step.

7. If after re-iteration there is no improvement on λ_i, stop. If it converges to the same estimate it means the local extremum corresponding to the bas-relief ambiguity is dominant and the normalized λ_i are the best estimate of the shape.

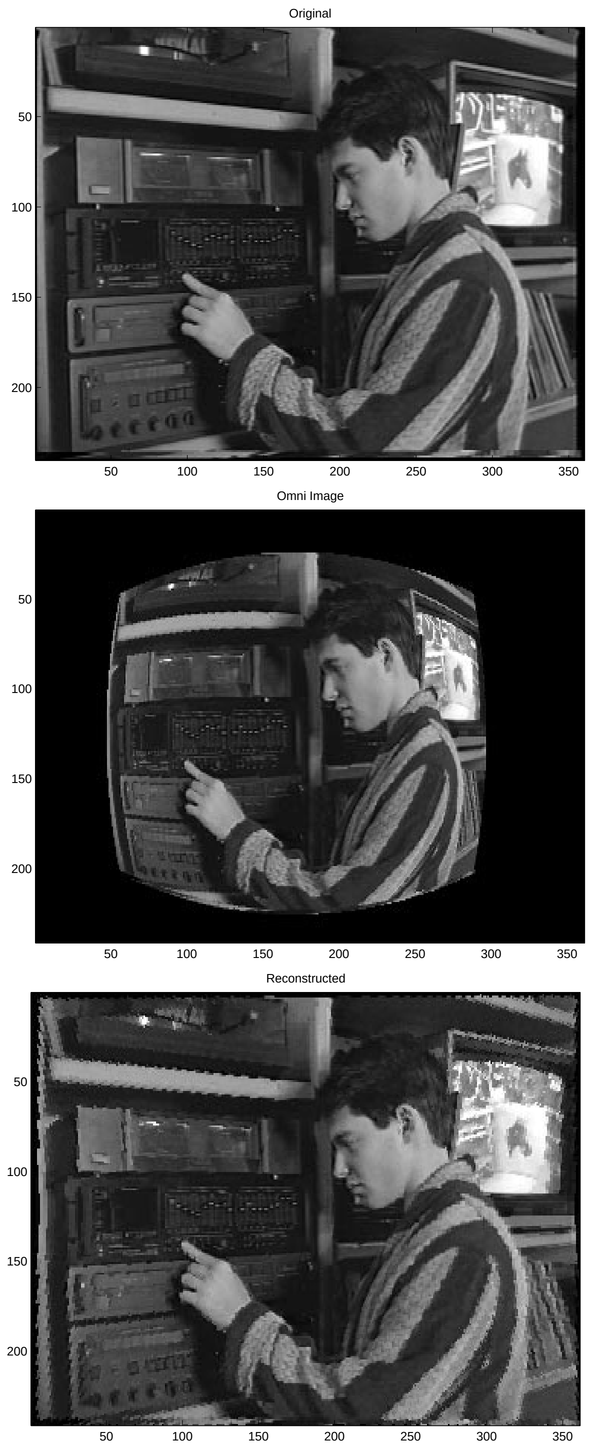

Figure 4.1: The original image is first projected to the omni-directional image plane and then re-projected back to the rectangular image plane.

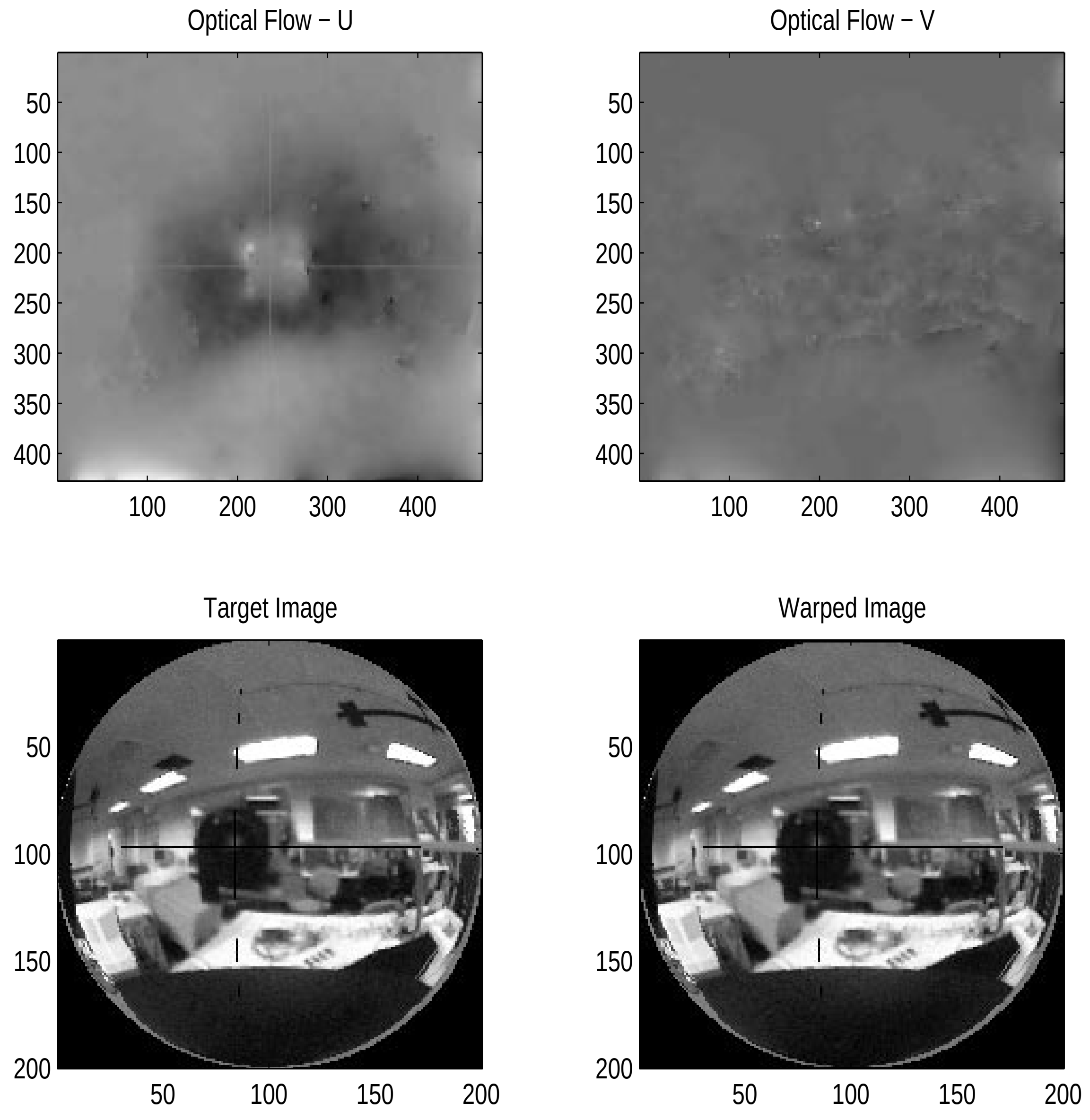

Figure 4.2: The rotation only sequence.

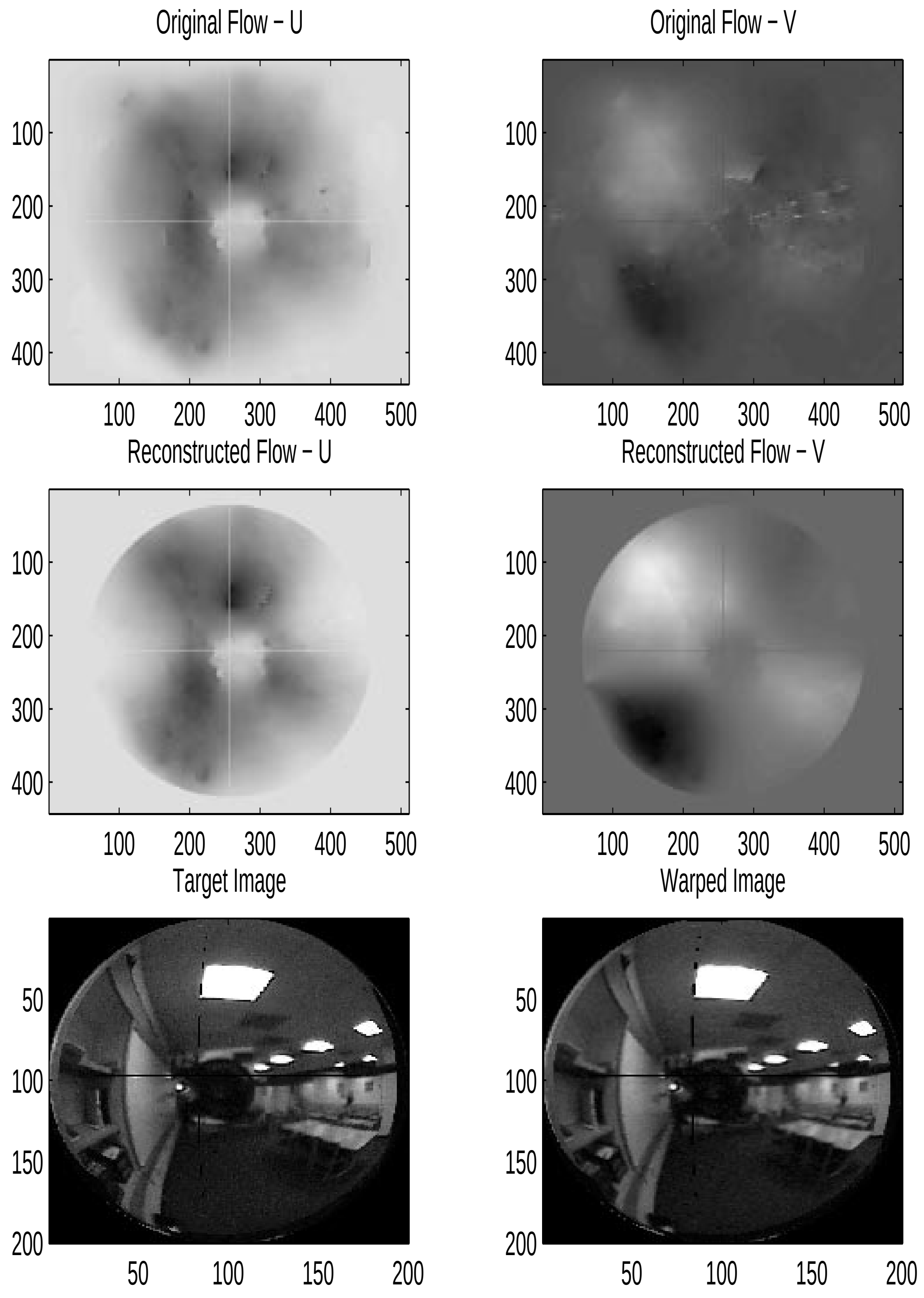

Figure 4.3: The sequence with both translational and rotational motion.

Chapter 5

Conclusion

Recent research proposed using omni-directional devices to capture motion instead of using conventional cameras. The large field of view of those omni-directional cameras facilitates more accurate motion estimation and enables applications with long baseline.

A few methods can be applied to compute motion parameters from omni-directional image sequences. One such way is to project the image and the optical flow onto a unit sphere and then do a bilinear constraint optimization. An alternative way is to map the image onto the unit sphere, then compute optical flow on the sphere and finally apply the optimization. Both methods have been tested in this project. Contradicting to [8], the latter method produced better result than the former. In addition, this project also proposed a way to perform direct estimate on the omni-direction image sequence without computing optical flow.

Future work on this project may include the correction of violations on brightness constancy assumption due to the fact that area change might be significant in non-linear projection. The direct method on the omni-directional plane would be another experiment that is worth to try.

Bibliography

[1] D. J. Heeger A. D. Jepson. Linear subspace methods for recovering translation direction. *Spatial Vision in Humans and Robots*, pages 39 – 62, 1993.

[2] M. J. Black A. Jepson. Mixture models for optical flow computation. *in Proc. Computer Vision and Pattern Recognition*, pages 760 – 761, 1993.

[3] N. Ahuja A. Krishnan. Panoramic image acquisition. *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 379 – 384, June 1996.

[4] B. K. Horn A. R. Bruss. Passive navigation. *Computer Vision, Graphics, and Image Processing*, 21:3 – 20, 1983.

[5] G. Adiv. Determing three-dimentional motion and structure from optical flow generated by several moving objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-7(4):384 – 401, July 1985.

[6] T. S. Huang B. L. Yen. Determining 3-d motion and structure of a rigid body using the spherical projection. *Computer Vision, Graphics, and Image Processing*, pages 21 – 32, 1983.

[7] S. E. Chen. Quicktime vr - an image based approach to virtual environment navigation. *Computer Graphics: Proc. of SIGGRAPH 95*, pages 29 – 38, August 1995.

[8] J. Gluckman and S. K. Nayar. Ego-motion and omnidirectional cameras. *Proc. of ICCV 98*, 1998.

[9] P. Anandan M. Black. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, (1):75 – 104, January 1996.

[10] S. Teller M. E. Antone. Automatic recovery of relative camera rotations for urban scenes. *Proc. CVPR*, pages II–282 – II–289, 2000.

[11] J. Santos-Victor N. Winters. Mobile robot navigation using omni-directional vision. *Proc. IMVIP 99*, 1999.

[12] V. Nalwa. A true omnidirectional viewer. *Tech Report, Bell Laboratories*, February 1996.

[13] S. K. Nayar. Catadioptric omnidirectional camera. *Computer Vision and Pattern Recognition*, 1997.

[14] S. K. Nayar. Omnidirectional video camera. *Proc. of DARPA Image Understanding Workshop*, May 1997.

[15] R. Brockett S. Soatto. Optimal structure from motion. *Proc. to IEEE Int. Conf. on Computer Vision*, April 1997.

[16] D. J. Heeger T. Y. Tian, C. Tomasi. Comparison of approaches to egomotion computation. *Computer Vision and Pattern Recognition*, 1996.

[17] R. C. Frecker W. J. MacLean, A. D. Jepson. Recovery of egomotion and segmentation of independent object motion using the em algorithm. *in Proc. of the 5th British Machine Vision Conference*, pages 13 – 16, 1994.