A Study in 3D Structure Detection Implementing Forward Camera Motion

Thesis for the M.Sc. Degree

By

D.M.Bappy

And

Md. Hamidur Rahman

Under the Supervision of

Dr. Siamak Khatibi

Department Of Electrical Engineering

Blekinge Institute Of Technology-Sweden

December, 2011

Examiner: Dr. Sven Johansson

Abstract

In this thesis we have studied detection of 3D structures having a forward camera movement which has strong influence of translation along the optical axis of the camera. During the forward movement the camera might undergoes rotation and translation .We have used “Plane plus Parallax” algorithm to cancel out this unwanted rotation. The input to the algorithm is a sequence of frames aligned with respect to a certain planar surface. The algorithm gives three types of outputs. (i) Dense correspondence across all frames. (ii) Dense 3D structure relative to the planar surface. (ii) Focus of Expansion (FOE) in all frames with respect to reference frame. Camera calibration is not needed for this algorithm. We have applied this algorithm to real world images and synthetic images. In both cases the 3D structure information could be obtained clearly even for objects far from the reference plane. Our result shows the potential of the method in 3D reconstruction implementing ego-motion of a single camera.

Acknowledgements

We would like to thank Dr.Siamak Khatibi to supervise us for this exciting thesis. He always helped us with new ideas and makes us understand the problem. We are also grateful to Dr. Michal Irani to cooperate with us. Finally, a special thanks to our family for always being with us.

Contents

List of Figures

Chapter 1

1 Introduction

In many applications, e.g. in cars or unmanned robots, a camera is used to detect depth and map the environment. This becomes possible if the camera movement is considered. This movement has normally strong influence of translation. When the translation is only along the optical axis of the camera, focus of expansion (FOE) point becomes an important issue. This point is at same position in the two images captured before and after the translation. However more generally during the practical movement of the camera along the optical axis rotation and other translation can as well occur where these kinds of transformation are not desired.

More generally a camera movement can be estimated using "feature based" methods, e.g. the methods, e.g. the method suggested by Torr, P.H.S. and Zisserman [1]. This method follows a strategy of concentrating on image regions which are information enriched. In such regions it is more likely to find correspondent points when we search in two subsequent images. From these correspondence an initial geometry is made. This geometry is used to guide further correspondences in less informative region of the images. Camera motion can also be estimated using optical flow [2, 3]. There are different motion models [4] depending on how to calculate the optical flow. All of these models need additional assumptions about the structure of the computed motion, due to under-constrained problem in the optical flow computation.

There are several methods which attempt to reduce or eliminate the undesired rotation and translation problem. The "plane+parallax" approach estimates the parallax displacements of a point between two views relative to a real or virtual planar surface in the scene which is called "reference plane" [2, 5, 6, 7, 8, 9]. The key concept behind this approach is that when the images are aligned with respect to planar surface, the rotation is cancelled out, the residual motion is only because of translational motion which contributes to the deviations of scene structure from the planar surfaces. The planar-parallax displacements form a radial flow field direction towards epipole [6, 7]. By applying this framework it is possible to recover scene structures relative to the reference plane. This method does not need calibrated camera or prior correspondence estimation-, and can be applicable directly to image brightness value. This algorithm computes a

dense 3D map of scene which is observed by the deviations of the scene from the reference plane. By using this algorithm it is possible to calculate both the epipole as well as the dense parallax field simultaneously.

The “plane+parallax” based algorithm works only for those images that are already aligned with a certain planar surface. This planar surface can be floor, wall in case of indoor scenes and ground plane, trees at far for the outdoor scenes. Prior alignment process is a drawback for this algorithm but still this algorithm has lots of benefits which make it interesting.

In this thesis we have used “plane+parallax” algorithm for multiple images rather than two images [6, 7]. Multi-frame algorithm overcomes so many problems which occurred by two-frame algorithm.

The goal of the thesis is to study movement of a camera which has strong influence of translation along the optical axis of the camera. Also to find key issues of a methodology to eliminate the effect of other transformations than translation along the optical axis, this can be rotation and other translations.

1.1 Thesis Outline

Chapter 2 describes shortly about the stereo vision and the special case of stereo vision called forward motion. Chapter 3 describes shortly the idea about parallax and derives “plane+parallax” decomposition. Chapter 4 discussed about a related method and little overview about our proposed method. Chapter 5 gives the idea about how the image registration is done with respect to planar surface. Chapter 6 presents the details of our proposed method. Chapter 7 introduces some results that we have got applying the proposed method. Chapter 8 presents the conclusion of our work.

Chapter 2

2 Stereo Vision

Stereo vision is the method by which we can determine 3D position of objects in a scene by comparing two images taken by two separate cameras or one camera in two different positions.

As we are working with the images it is always better to know the way they are formed. So the mathematical derivation for the camera model is discussed.

2.1 Model of Camera

The most basic and important camera model is called pinhole camera model. It describes the relationship between the coordinates of a 3D point and its projection to the image plane. Geometric distortion is not included with this model. This model can only be used as a first order approximation of the mapping from a 3D scene to a 2D image. The camera model is defined by a projection centre and an image plane (figure 2.1.1).

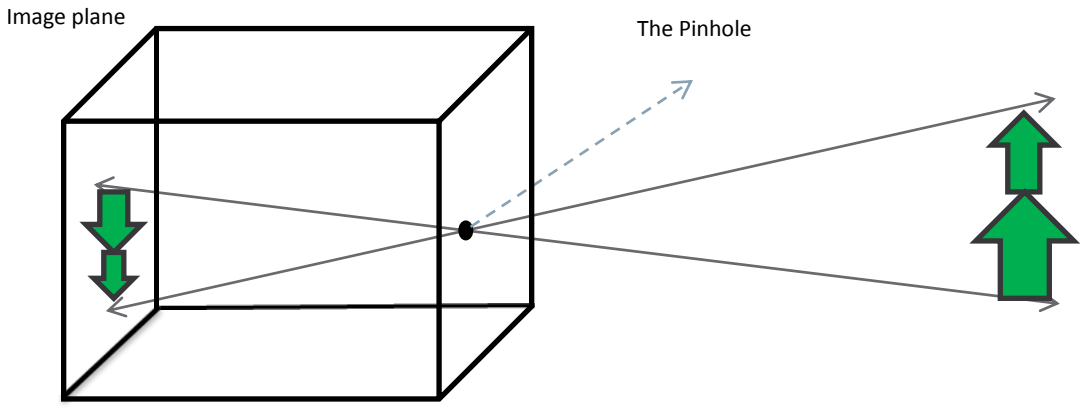

Figure 2.1.1: Pinhole Camera model

Figure 2.1.2 gives us clear idea about the terms related to this camera model. The distance between the projection centre and the image plane is called focal length. The line passing through the projection centre is called optical axis. The intersection point of optical axis and the image plane is called principal point. Principal plane is the plane that is parallel to the image plane and holds the centre of projection.

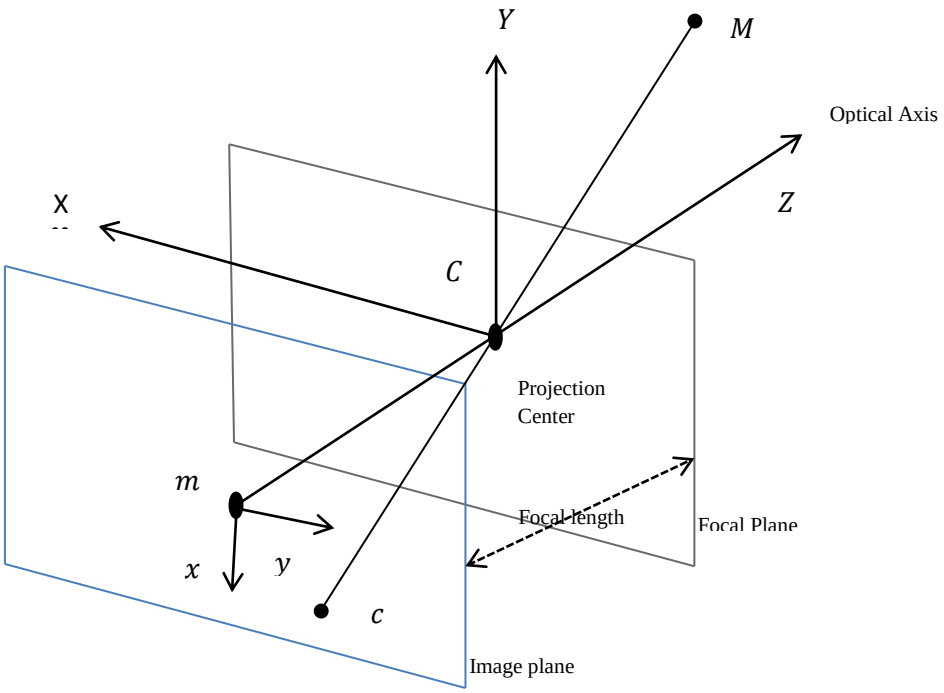

Figure 2.1.2 : Pinhole Camera Model Geometry

When the origin of the coordinate system is placed on the projection center then the x-y plane parallel to the image plane and optical axis is aligned with the Z-axis.

If a 3D point M with coordinates (X, Y, Z) is projecting on to image plane with coordinates (x, y) then Thales theorem is applicable on the triangle in figure 2.1.3 which gives

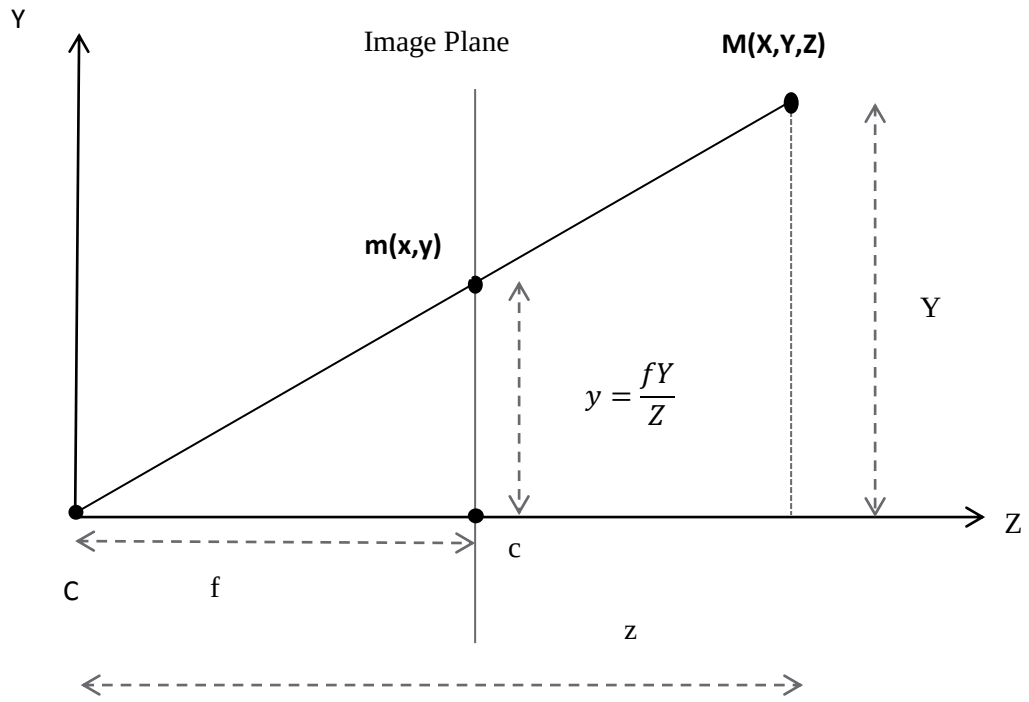

Figure 2.1.3: The Projection on Camera Model onto YZ

$$\frac{y}{Y} = \frac{f}{Z}\text{, similarly } \frac{x}{X} = \frac{f}{Z} \qquad [2.1]$$

All points on the line CM projects on the image point m which is equivalent to rescaling of point represented in homogeneous coordinates.

$$y = f\frac{Y}{Z} = f\frac{sY}{sZ}\text{, similarly } x = f\frac{X}{Z} = f\frac{sX}{sZ} \qquad [2.2]$$

2.2 Projection Matrix of Camera

Equation 2.2 can be written in the matrix form where the world and image coordinates are expressed by homogeneous coordinate system.

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad [2.3]$$

In equation 2.3 s represents the scaling factor and f is the focal length. Now 3D point M (X,Y,Z) and the corresponding projected image point m (x,y) can be denoted in homogeneous coordinate as $\widetilde{M}$ and $\widetilde{m}$ respectively then the equation 2.3 becomes

$$s\widetilde{m} = P\widetilde{M} \qquad [2.4]$$

where P is the perspective projection matrix.

So far our coordinate system is assigned to project center. However we need to present any 3D point in an arbitrary world coordinate system.

Figure 2.2.1 shows the transformation from the camera (C) to the world (O) coordinate system. The rotation R_{3x3} followed by the translation t_{3x1} describes the orientation and position of the camera in the world coordinate system. These translation and rotation parameters are also called extrinsic parameters.

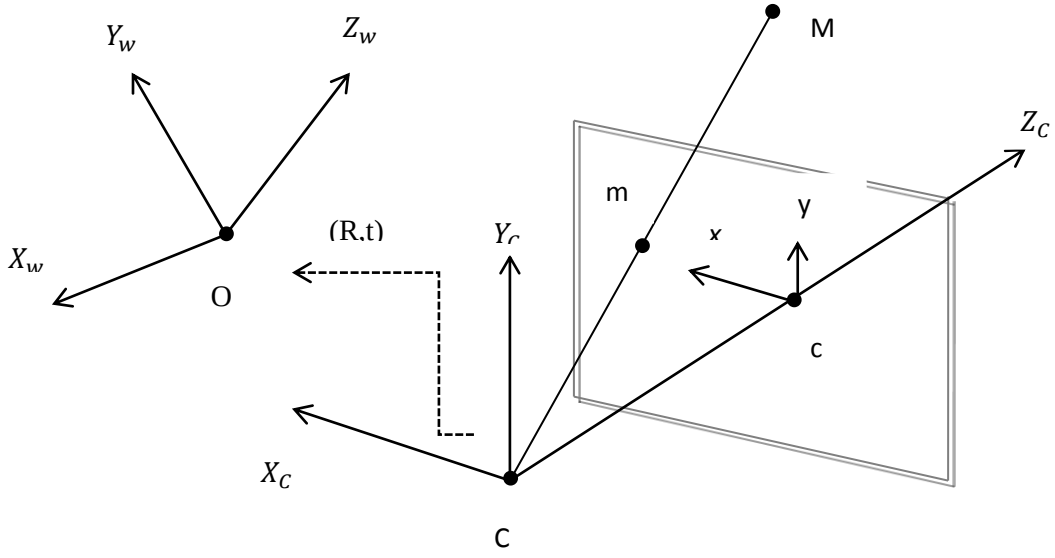

Figure 2.2.1 : Camera, world and image plane coordinates system are demonstrated

A point M_C in camera coordinate system is related to a corresponding point M_W in world coordinate system:

$$M_C = R(M_W + t) \qquad [2.5]$$

$$\text{Where } M_C = \begin{bmatrix} X_C \\ Y_C \\ Z_C \end{bmatrix} \quad and \quad M_W = \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$$

Or in homogenous coordinates :

$$\breve{M}_C = G\breve{M}_W \qquad [2.6]$$

Where the matrix G_{4x4} is

$$G = \begin{bmatrix} R & t \\ 0_3^T & 1 \end{bmatrix} \qquad [2.7]$$

Combining the equation 2.4 and 2.6 gives us

$$m \sim PM_C = PGM_W = P_{new}M_W \qquad [2.8]$$

In reality the origin of the image coordinate is not the principal point as we have known from the pinhole camera model and also the scaling corresponding to image axis is different. In case of CCD camera these depend on the size and shape of the pixels. So the coordinates in the image plane need to transform by multiplying the matrix P to the left by 3x3 matrix K_1. Figure 2.2.2 shows the relation between pixel coordinates and image coordinates. Now the camera perspective model is:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim K_1 . \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} . \begin{bmatrix} R & t \\ 0_3^T & 1 \end{bmatrix} . \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad [2.9]$$

where K_1 is defined as

$$K_1 = \begin{bmatrix} k_u & k_v cot\theta & u_0 \\ 0 & k_v / sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad [2.10]$$

parameters k_u and k_v denotes the scaling factor of the axes of image plane,θ is the skew between axes and (u_0, v_0) is the principal point. This matrix does not depend on the camera orientation and position. The parameters inside this matrix are called intrinsic parameters.

Setting the values of K_1 in equation 2.9 we get

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} k_u & k_v cot\theta & u_0 \\ 0 & k_v/sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} R & t \\ 0_3^T & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \leftrightarrow$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} fk_u & fk_v cot\theta & u_0 & 0 \\ 0 & fk_v/sin\theta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} R & t \\ 0_3^T & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \leftrightarrow$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} \alpha_u & \alpha_v cot\theta & u_0 \\ 0 & \alpha_v/sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} R & t \\ 0_3^T & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad [2.11]$$

where $\alpha_u = fk_u$, $\alpha_v = fk_v$ and

$$K_2 = \begin{bmatrix} fk_u & fk_v cot\theta & u_0 & 0 \\ 0 & fk_v/sin\theta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad [2.12]$$

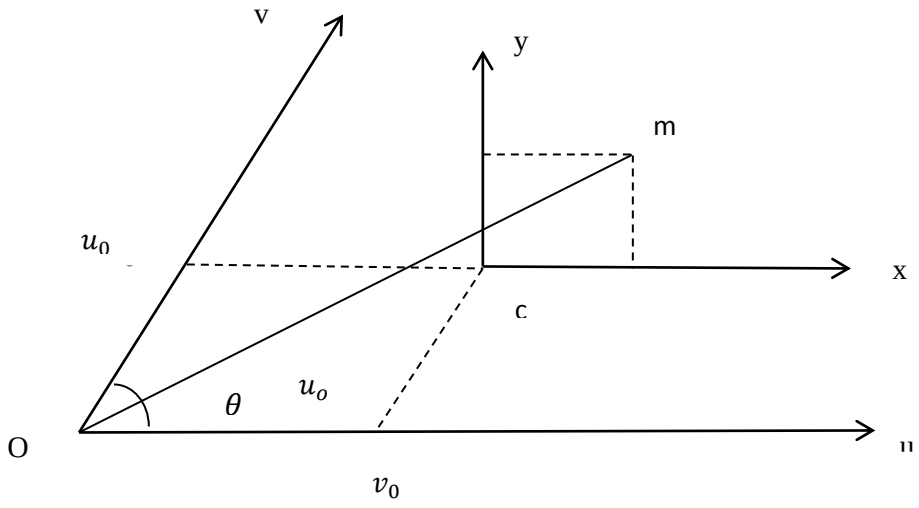

Figure 2.2.2 : Relation between pixel coordinates and image

2.3 Geometry of Two-view

In Two-view geometry the relation between a 3D scene point and its projection onto the 2D image points is discussed. Normally two cameras view a 3D scene from two distinct positions and epipolar geometry describes a number of geometric characteristics in 2D view. These geometric properties are known as epipolar constraint in computer vision.

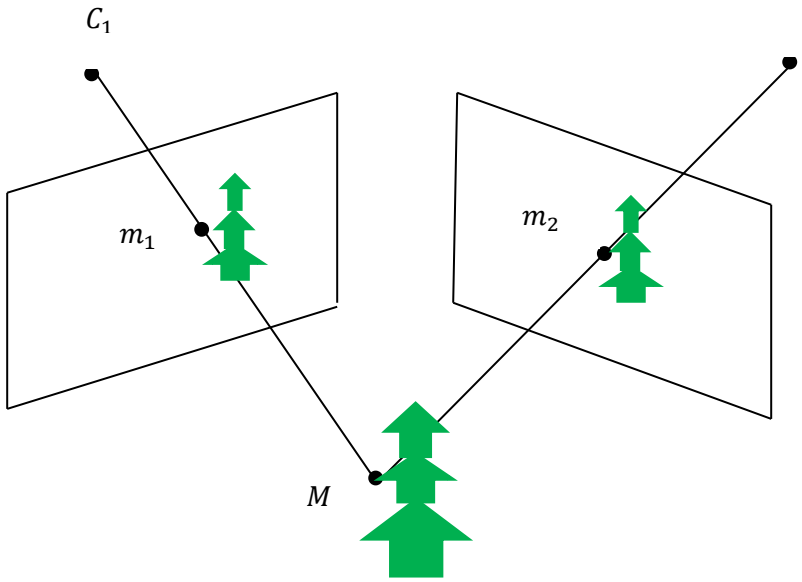

Figure 2.3.1 : Corresponding points in two views of the same scene

The determination of the scene position of an object point depends upon matching the image location of the object point in one image to the location of the same object point in the other image. The process of establishing such matches between points m_1 and m_2 in a pair of images is called correspondence.

At first it might seem that correspondence requires a search through the whole image, but the epipolar constraint reduces this search to a single line. To see this, we consider the following figure(Figure 2.3.2):

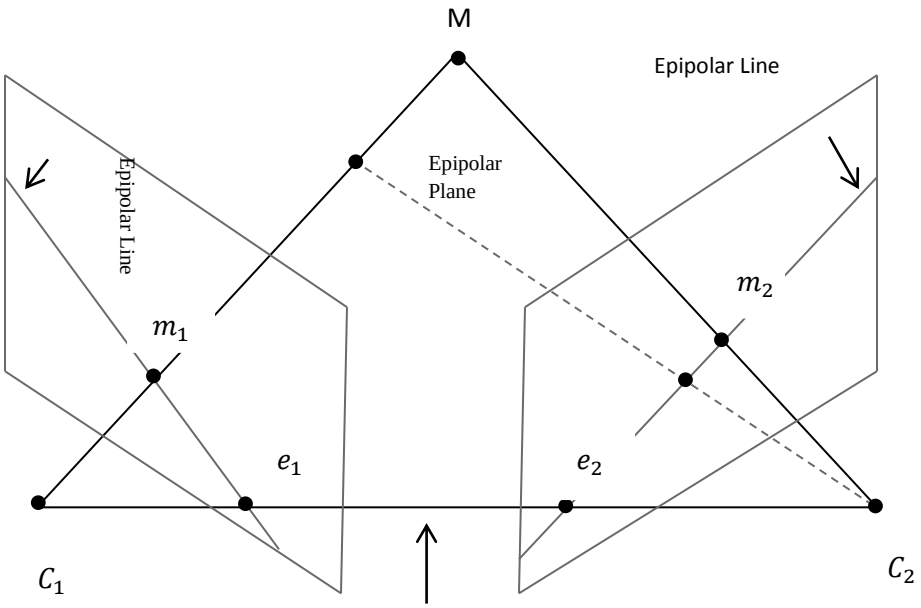

Figure 2.3.2 : Epipolar geometry and the epipolar constraint

The epipole e_1 or e_2 is the point of intersection of the line joining the optical centres C_1 andC_2, that is the baseline, with the image plane. Thus the epipole is the image, in one camera, of the optical centre of the other camera.

The epipolar plane is the plane defined by a 3D point M and the optical centers C_1 andC_2.

The epipolar lines l_1 and $l_2$are the straight line of intersection of the epipolar plane with the image plane. It is the image in one camera of a ray through the optical centre and image point in the other camera. All epipolar lines intersect at the epipole.

The equation of the optical ray going through a projected point m is obtained to get the equation of epipolar line. The optical ray is formed by the line passing through projection centre C and projected point m. If we choose D as a point on the ray then,

$$\widetilde{m} = P \begin{bmatrix} D \\ 1 \end{bmatrix} \qquad [2.13]$$

where P is a 3x4 matrix with $[B_{3x3} \quad b_{3x1}]$

Now equation 2.13 can be written as $\widetilde{m} = [B_{3x3} \quad b_{3x1}] \begin{bmatrix} D \\ 1 \end{bmatrix}$, and the 3D point D can be calculated as:

$$D = B^{-1}(-b + \widetilde{m}) \qquad [2.14]$$

A point on the optical ray can be represented as

$$\text{M}= C + \delta(D - C) = B^{-1}(-b + \widetilde{m}) \qquad [2.15]$$

where $\delta \in (0, \infty)$, or

$$\widetilde{M} = \begin{bmatrix} -B^{-1} \\ 1 \end{bmatrix} + \delta \begin{bmatrix} B^{-1}\widetilde{m} \\ 0 \end{bmatrix} \qquad [2.16]$$

If we assume P_1 and P_2 be the projection matrices of two cameras which corresponds to two views, and m_1 be the projected point on the first image plane than the projection of the optical ray going through the point m_1 on the second image plane introduce us the corresponding epipolar line. This can be represented as:

$$s_2\widetilde{m_2} = P_2\widetilde{M} = P_2 \begin{bmatrix} -{B_1}^{-1}b_1 \\ 1 \end{bmatrix} + \delta_1 P_2 \begin{bmatrix} {B_1}^{-1}\widetilde{m_1} \\ 0 \end{bmatrix} \qquad [2.17]$$

The equation of the epipolar line l_2 can be represented as :

$$s_2\widetilde{m_2} = e_2 + \delta_1 B_2 {B_1}^{-1}\widetilde{m_1} \qquad [2.18]$$

The equation 2.18 describes the geometrical relation between two views in terms of projection matrices and assumes that intrinsic and extrinsic parameters are known.

Our work in this thesis is mainly dependent on the special case of stereo vision. Figure 2.3.3 shows that the camera is moving forward along optical axis. It doesn't have any rotation only pure translation occurred in this case. Two camera centers C, C' and epipoles e, e' are on the same line.

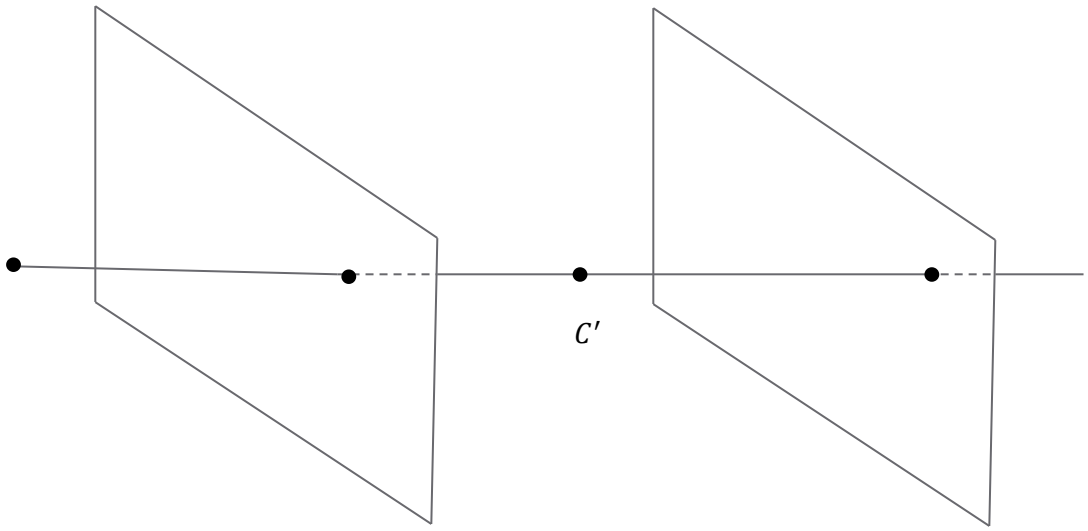

Figure 2.3.3: Pure camera translation

When the camera is translating along optical axis without any rotation then the epipoles for two images have the same coordinates. All other points looks like radiating from the epipole along lines. The epipole in figure 2.3.4 is called Focus of Exapnsion (FOE) for this special case.

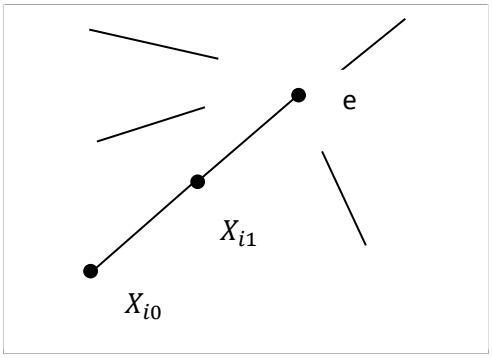

Figure 2.3.4: Focus of Expansion

Chapter 3

3 Motion Parallax

Let us consider we have two 3D points which are captured by two positions of one camera. Let us also assume the base line between the two camera positions is normal to the optical axis of the camera in each position, i.e. we have only lateral translation between cameras. If our 3D points have different depth in relation to the base line, the relative displacement or motion of the 3D points resulting from the lateral movement is called motion parallax. Normally in capturing of a scene our interest is to have certain objects in focus or in other words we are considering a fixation point. A proximal point that moves with the two camera movement appears to be behind the fixation point, and a proximal point that moves against the camera movement appears to be closer than the fixation point. Figure 3.1 gives clear idea about the parallax motion.

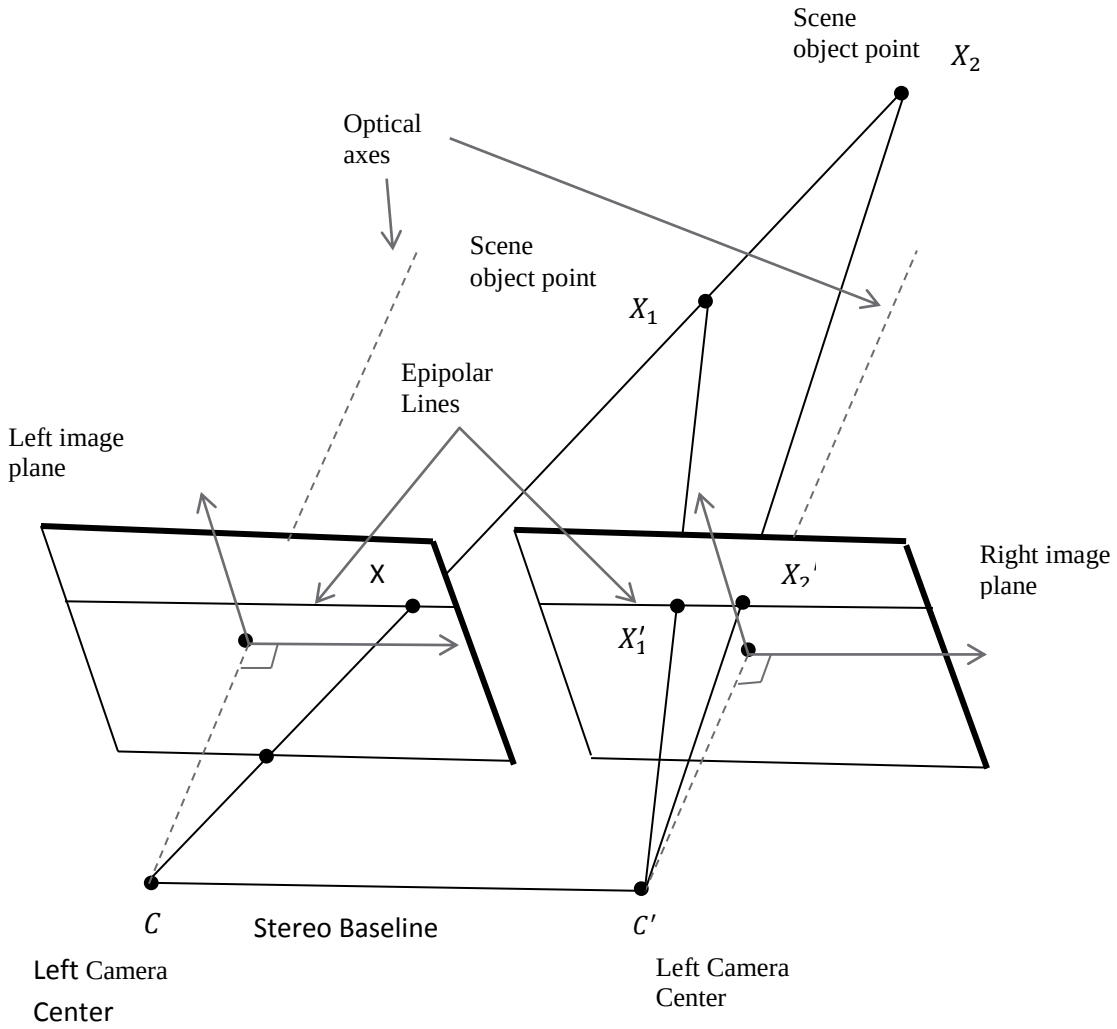

Figure 2.3.1: Motion Parallax

Figure 3.1 shows the lateral translation of the camera. Two 3D points X_1 and X_2 has been captured from the two positions of the camera. The images of the 3D points $X_1$and X_2 are coincident at the time viewed by the camera with centre C. The images of the 3D points are not coincident when it is viewed by the camera centre C'. Because the camera centre C' doesn't lie along the line L that goes through X_1 and X_2. The image points x_1' and x_2' gives a line which is actually the image of the ray represents the amount of parallax.

3.1 Plane Induced Parallax

The world plane (denoted as π) introduces a homography H between the two images, as a result the image of points on the plane are mapped as $x' = Hx$, where x and x' are the image points in the first and second views respectively. The homography can be calculated for a minimum of four correspondences across the two views of points (or lines) on the plane [10]. After that the fundamental matrix for the views is given by [11].

$$F = H^{-T}[e]_x = [e']_x H$$

where e' denote the epipole in the second image and $[e']_x$ is the $3x3$ skew matrix where as $[e]_x x = [e']_x x$. The vector joining the image of a world point x' with the transferred image of that point from the first view $\tilde{x} = Hx$ is called parallax vector.

The plane π in a static scene where a 3D point X intersects the plane at the point X_π. 3D points X and X_π have images as coincident points at x in the first view. In the second

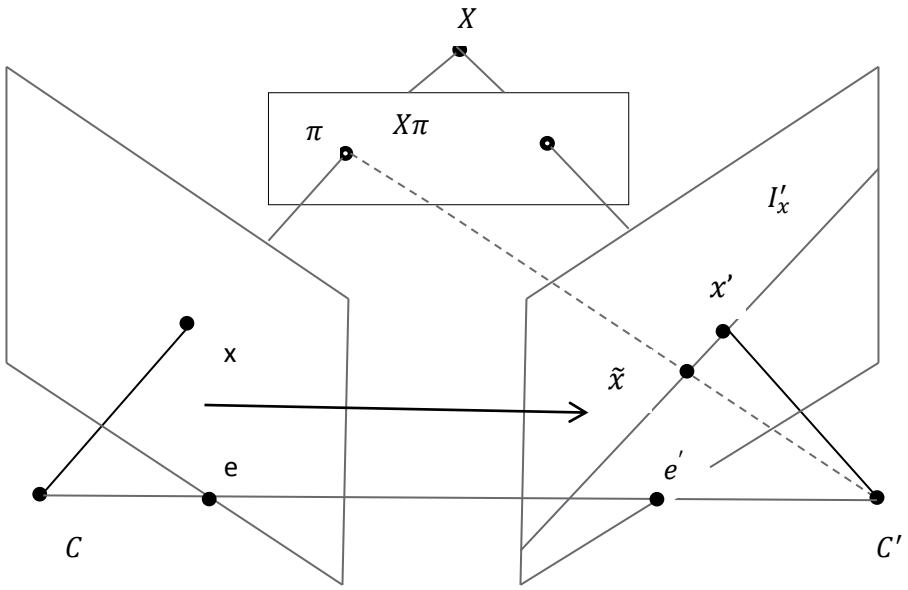

Figure 3.1.1: Plane induced parallax

image the points are x' and $\tilde{x}$'. These points are not coincident because X is not on the plane. The vector between x' and $\tilde{x}$' is called parallax relative to the plane π.

3.2 Plane plus Parallax

The plane plus parallax geometry is discussed below which is the main idea of the method that we have used. The 2D image motion of a 3D scene point which is introduced between two images can be decomposed into two components [2,5,6,7,8,9,12] : (i) the image motion of a planar surface (i.e.,homohraphy), and (ii) the residual image motion known as " planar parallax". This decomposition can be described as below.

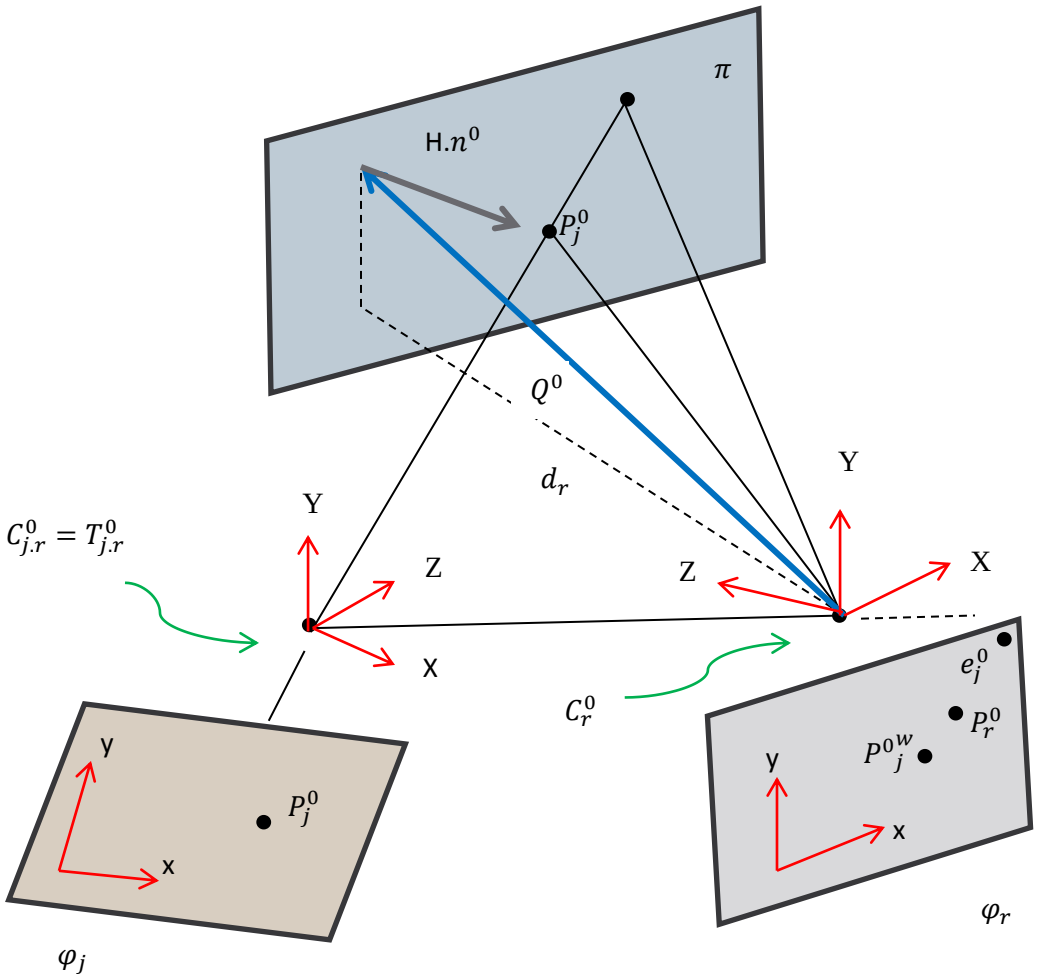

Figure 3.2.1 : Two-view planar parallax for Plane+Parallax Decomposition

Let us assume (X,Y,Z) be a Cartesian system with the origin at camera center and $\boldsymbol{P}^0 = (X^0 \ Y^0 \ Z^0)^T$ be a 3D scene point which projects onto an image point $\boldsymbol{p}^0 = (x^0 \ y^0)^T$. We have used the subscript "o" to denote Euclidean quantities. Let $\mathbf{P} \cong (X\ Y\ Z\ W)^T$ represents the point in 3D space (P_3) and $\mathbf{p} \cong (x\ y\ w)^T$ represents the point in 2D space (P_2). If $\boldsymbol{P^0 and\ p^0}$ are finite, and W≠ 0 and w≠ 0, then $\boldsymbol{P}^0 = \frac{\boldsymbol{P}}{W}$, $\boldsymbol{p}^0 = \frac{\boldsymbol{p}}{w}$ which represents the homogeneous vectors. Assume K is the 3×4 projection matrix that we obtained from equation 2.12 (bold fonts denote matrices):

$$\mathbf{p} \cong \boldsymbol{KP}$$

where≅ denotes equality up to a scale factor, and :

$$K = \begin{bmatrix} -f\alpha_x & -f\alpha_x cot\theta & -fx_0 0 \\ 0 & -f\alpha_y csc\theta & -fy_0 0 \\ 0 & 0 & 1 \quad 0 \end{bmatrix}$$

The matrix K depend on the camera intrinsic parameters: α_x and α_y are the horizontal and vertical scale factors that relate world units to pixel units, x_0 and y_0 are offsets of the optical axis from the center of image, θ is the angle between the x and y axes, and f is the focal length.

We assume $\{\Phi_j\}_{j=1}^{l}$ be a collection of images of a rigid scene taken from different viewing positions. $\{C_j\}_{j=1}^{l}$ denotes the corresponding camera centers. Different projection matrix K_j is associated with each image $\{\Phi_j\}_{j=1}^{l}$. A 3D scene point $\boldsymbol{P}^0{}_j$ in the coordinate system of camera j (where j = 1, …l) is projected onto image Φ_j at a pixel $\boldsymbol{p}^0{}_j$.

Let $\pi \in R^3$ be a planar surface in the scene. We represent π as the "reference plane". Let us assume all camera centers are located on one side of the plane. Let $\boldsymbol{n}^0{}_j$ represents the normal of the plane π in the coordinate system of camera j. The normal of the plane is defined to have a positive Z coordinate in all 3D coordinate systems. The height of a 3D scene point $\boldsymbol{P}^0$ from the plane, represented by H $\stackrel{\text{def}}{=} H(\boldsymbol{P}^0, \pi)$ which is negative if it is oriented towards the cameras, or positive otherwise (the same direction as the normal).The transformation of the coordinate system from a 3D coordinate system r to another 3D coordinate system j, is captured by:

$$\boldsymbol{p}^0{}_j = R_{j,r}\boldsymbol{p}^0{}_r + \boldsymbol{T}^0{}_{j,r} \qquad [3.1]$$

Where $R_{j,r}$ is an orthonormal rotation matrix and $\boldsymbol{T}^0{}_{j,r}$ is a translation vector, capturing the extrinsic camera parameters between these two camera views. We assume Φ_r be one of the images, chosen to be a "reference image" or "reference frame". We begin our analysis with respect to the 3D Euclidean Coordinate system of the reference frame, Φ_r. Let us assume the existence of a reference plane π, in the scene. $\boldsymbol{n}^0{}_r \stackrel{\text{def}}{=} n(\pi, r)$ is the normal of the reference plane π given with respect to the reference frame coordinate system. A 3D scene point can be present as a vector sum of the form:

$$\boldsymbol{p}^0{}_r = \boldsymbol{Q}^0 + \boldsymbol{H}\boldsymbol{n}^0{}_r$$

Where H represents the height of the point $\boldsymbol{p}^0{}_r$ over the plane (along the normal direction $\boldsymbol{n}^0{}_r$) and $\boldsymbol{Q}^0 \stackrel{\text{def}}{=} \boldsymbol{Q}^0(\pi, \boldsymbol{p}^0{}_r, r)$ represents the perpendicular projection of the point $\boldsymbol{p}^0{}_r$ on the plane π.

The inner product between a 3D scene point $\boldsymbol{p}^0{}_r$and a plane normal $\boldsymbol{n}^0{}_r$ yields:

$$\boldsymbol{n}^{0T}_r \boldsymbol{p}^0{}_r = \boldsymbol{n}^{0T}_r \boldsymbol{Q}^0 + H = d_r + H \qquad [3.2]$$

Where $d_r \stackrel{\text{def}}{=} \boldsymbol{n}^{0T}_r \boldsymbol{Q}^0$ is the perpendicular distance of the reference camera center $\boldsymbol{C}^0{}_r$ from the reference plane π, i.e. $d_r \neq 0$. Then from equation 3.1 and 3.2, the 3D coordinates of $\boldsymbol{p}^0{}_j$ in the coordinate system of camera $\boldsymbol{C}^0{}_j$ are:

$$\boldsymbol{p}^0{}_j = R_{j,r}\boldsymbol{p}^0{}_r + \boldsymbol{T}^0{}_{j,r}.1 = R_{j,r}\boldsymbol{p}^0{}_r + \boldsymbol{T}^0{}_{j,r}.1\left(\frac{\boldsymbol{n}^{0T}_r \boldsymbol{p}^0{}_r - H}{d_r}\right)$$

Where r denotes the index of the reference frame and j$\in \{1,2,\ldots,l\}\backslash\{r\}$ denotes the index of the other frames. Regrouping terms yields:

$$\boldsymbol{p}^0{}_j = \left(R_{j,r} + \frac{\boldsymbol{T}^0{}_{j,r}\boldsymbol{n}^{0T}_r}{d_r}\right)\boldsymbol{p}^0{}_r - \frac{H}{d_r}\boldsymbol{T}^0{}_{j,r} = A\boldsymbol{p}^0{}_r - \frac{H}{d_r}\boldsymbol{T}^0{}_{j,r}$$

Where: $A \stackrel{\text{def}}{=} A(j,r) = R_{j,r} + \frac{\boldsymbol{T}^0{}_{j,r}\boldsymbol{n}^{0T}_r}{d_r}$ represents a 3×3 matrix corresponding to the 3D Euclidean transformation matrix of the plane π between the coordinate system of $\boldsymbol{C}^0{}_r$ and $\boldsymbol{C}^0{}_j$.

The projection of $\boldsymbol{p}^0{}_j$ into frame j in homogenous coordinate is:

$$P_j = \lambda_1 K_j \boldsymbol{p}^0{}_j = \lambda_1 K_j \boldsymbol{A}\boldsymbol{p}^0{}_r - \frac{H}{d_r}\lambda_1 K_j \boldsymbol{T}^0{}_{j,r}$$

where λ_1 is a non-zero scalar. Similarly:$P_r = \lambda_2 K_r \boldsymbol{p}^0{}_r$, i.e. $\boldsymbol{p}^0{}_r = \frac{1}{\lambda_2}\boldsymbol{K}^{-1}{}_r P_r$

Hence

$$P_j = \lambda_1 K_j A \boldsymbol{K}^{-1}{}_r \frac{1}{\lambda_2} P_r - \frac{H}{d_r}\lambda_1\left(\frac{1}{\lambda_3}\lambda_3\right)K_j \boldsymbol{T}^0{}_{j,r} = BP_r - \frac{H}{d_r}\left(\frac{\lambda_1}{\lambda_3}\right) \qquad [3.3]$$

where B$\stackrel{\text{def}}{=} B(j,r) = \frac{\lambda_1}{\lambda_2} K_j A \boldsymbol{K}^{-1}{}_r$ denotes a homography (i.e. a 2D projective transformation that is an invertible 3-matrix) capturing the coordinate transformation of the plane π between views $\Phi_r$and Φ_j, and e = $(\mathrm{e}_1\, \mathrm{e}_2 \mathrm{e}_3)^T \stackrel{\text{def}}{=} e(j,r) = \lambda_3 K_j \boldsymbol{T}^0{}_{j,r}$ is the projection of camera center $\boldsymbol{C}^0{}_r$ on image Φ_j (i.e. the epipole), and λ_3 is a non-zero scalar. Note that for points $\boldsymbol{p}^0{}_r$ on the reference plane π (i.e., H = 0): $P_j = BP_r$

Let:

$$\boldsymbol{p}^w{}_j \stackrel{\text{def}}{=} \boldsymbol{B}^{-1} P_j \qquad [3.4]$$

From equations 3.3 and 3.4:

$$\boldsymbol{p}^w{}_j \stackrel{\text{def}}{=} \lambda_4 \boldsymbol{B}^{-1} P_j = \lambda_4 P_r - \frac{H}{d_r} \frac{\lambda_4 \lambda_1}{\lambda_3 \lambda_5} \boldsymbol{e}^w$$

Or:

$$\boldsymbol{p}^w{}_j - \lambda_4 P_r = -\frac{H}{d_r} \frac{\lambda_4 \lambda_1}{\lambda_3 \lambda_5} \boldsymbol{e}^w \qquad [3.5]$$

Where $\boldsymbol{e}^w = (\mathrm{e}_1{}^w \mathrm{e}_2{}^w \mathrm{e}_3{}^w)^T \stackrel{\text{def}}{=} \lambda_5 \boldsymbol{B}^{-1} e$ represents the warped epipole e and λ_4, $\lambda_5$are non zero scalars. Note that for points on the plane π (i.e. H = 0):

$\boldsymbol{p}^w{}_j = \boldsymbol{p}_j$

because $\boldsymbol{p}^w{}_j$and $\boldsymbol{p}_j$ belongs to actual image points and not to points at infinity, it follows that $\boldsymbol{w}_r \neq 0$and $\boldsymbol{w}^w{}_j \neq 0$. Using the fact that $\boldsymbol{p}^0{}_r = \frac{P_r}{w_r}$ and $\boldsymbol{p}^0{}^w_j = \frac{\boldsymbol{p}^w{}_j}{\boldsymbol{w}^w{}_j}$.

Equation 3.5 can be re-written as

$$\boldsymbol{w}^w{}_j \boldsymbol{p}^0{}^w_j - \lambda_4 \boldsymbol{w}_r \boldsymbol{p}^0{}_r = -\frac{H}{d_r} \frac{\lambda_4 \lambda_1}{\lambda_3 \lambda_5} \boldsymbol{e}^w \qquad [3.6]$$

dividing the last equation by $\lambda_4 \boldsymbol{w}_r$ yields:

$$\frac{\boldsymbol{w}^w{}_j \boldsymbol{p}^0{}^w_j}{\lambda_4 \boldsymbol{w}_r} - \boldsymbol{p}^0{}_r = -\frac{H}{\boldsymbol{w}_r d_r} \frac{\lambda_4 \lambda_1}{\lambda_3 \lambda_5} \boldsymbol{e}^w \qquad [3.7]$$

From substituting the third component of 3.7, it can be obtained, that:

$$\frac{\boldsymbol{w}^{w}{}_{j}}{\lambda_4 \boldsymbol{w}_r} = 1 - \frac{\lambda_1}{\lambda_3\lambda_5}\frac{H}{w_r}\frac{\mathrm{e}_3{}^{w}}{d_r}.$$

Setting this result into 3.7 and using $w_r = \lambda_2 \boldsymbol{Z}^0{}_r$, gives:

$$\left(1 - \frac{\lambda_1}{\lambda_2\lambda_3\lambda_5}\frac{H}{\boldsymbol{Z}^0{}_r}\frac{\mathrm{e}_3{}^{w}}{d_r}\right)\boldsymbol{p}^{0}{}^{w}_{j} - \boldsymbol{p}^0{}_r = -\frac{H}{\boldsymbol{Z}^0{}_r}\frac{1}{d_r}\frac{\lambda_1}{\lambda_2\lambda_3\lambda_5}\mathrm{e}_3{}^{w}$$

Reconstructing terms yields:

$$\boldsymbol{u}^{0}{}^{w}_{j} \overset{\text{def}}{=} \boldsymbol{p}^{0}{}^{w}_{j} - \boldsymbol{p}^0{}_r = \frac{H}{\boldsymbol{Z}^0{}_r}\left(\frac{\lambda_1}{\lambda_2\lambda_3\lambda_5}\frac{\mathrm{e}_3{}^{w}}{d_r}\boldsymbol{p}^{0}{}^{w}_{j} - \frac{\lambda_1}{\lambda_2\lambda_3\lambda_5}\frac{\mathrm{e}_3{}^{w}}{d_r}\right) \qquad [3.8]$$

where $\boldsymbol{u}^{0}{}^{w}_{j} = (\boldsymbol{u}^{0}{}^{w}_{j}\boldsymbol{v}^{0}{}^{w}_{j})^T$ represents the residual image motion and is denoted to as the "planar parallax" displacement. Equation 3.8 can be written more compactly as:

$$\boldsymbol{u}^{0}{}^{w}_{j} \overset{\text{def}}{=} \boldsymbol{p}^{0}{}^{w}_{j} - \boldsymbol{p}^0{}_r = \gamma(\mathrm{e}_3{}^{w}\boldsymbol{p}^{0}{}^{w}_{j} - \mathrm{e}^{w}) \qquad [3.9]$$

where γ=$\gamma\left(\boldsymbol{p}^0{}_r\right) \overset{\text{def}}{=} \frac{H}{\boldsymbol{Z}^0{}_r}$and $\mathrm{e}^{w} = (\mathrm{e}_1{}^{w}\mathrm{e}_2{}^{w}\mathrm{e}_3{}^{w})^T \overset{\text{def}}{=} \frac{\lambda_1}{\lambda_2\lambda_3\lambda_5}\frac{\mathrm{e}^{w}}{d_r} \cong \mathrm{e}^{w}$.

We want to express $\boldsymbol{u}^{0}{}^{w}_{j}$ in terms of $\boldsymbol{p}^0{}_r$ (as opposed to $\boldsymbol{p}^{0}{}^{w}_{j}$, which is unknown).

This is necessary for multi frame estimation. Regrouping terms of equation 3.9 yields:

$$(1 - \gamma\mathrm{e}_3{}^{w})\boldsymbol{p}^{0}{}^{w}_{j} - \boldsymbol{p}^0{}_r = -\gamma\mathrm{e}^{w} \qquad [3.10]$$

Including $\gamma\mathrm{e}_3{}^{w}\boldsymbol{p}^0{}_r$ to both sides:

$$(1 - \gamma\mathrm{e}_3{}^{w})\boldsymbol{p}^{0}{}^{w}_{j} - \boldsymbol{p}^0{}_r + \gamma\mathrm{e}_3{}^{w}\boldsymbol{p}^0{}_r = -\gamma\mathrm{e}^{w} + \gamma\mathrm{e}_3{}^{w}\boldsymbol{p}^0{}_r$$

Or:

$$(1 - \gamma\mathrm{e}_3{}^{w})\boldsymbol{p}^{0}{}^{w}_{j} + (\gamma\mathrm{e}_3{}^{w} - 1)\boldsymbol{p}^0{}_r = -\gamma\mathrm{e}^{w} + \gamma\mathrm{e}_3{}^{w}\boldsymbol{p}^0{}_r$$

Adding common terms:

$$(1-\gamma \mathrm{e_3}^w)(\boldsymbol{p}^0{}_j^w - \boldsymbol{p}^0{}_r) = \gamma(\mathrm{e_3}^w \boldsymbol{p}^0{}_r - \mathrm{e}^w)$$

Dividing equation (13) by$(1-\gamma \mathrm{e_3}^w)$ gives the desired expression for the parallax displacement:

$$\boldsymbol{u}^0{}_j^w = \boldsymbol{p}^0{}_j^w - \boldsymbol{p}^0{}_r = \frac{\gamma}{1-\gamma \mathrm{e_3}^w}\left(\mathrm{e_3}^w \boldsymbol{p}^0{}_r - \mathrm{e}^w\right) \qquad [3.11]$$

So, the " plane+parallax" decomposition of image motion $\boldsymbol{u}^0{}_j$ is:

$$\boldsymbol{u}^0{}_j = \boldsymbol{p}^0{}_j - \boldsymbol{p}^0{}_r = \left(\boldsymbol{p}^0{}_j - \boldsymbol{p}^0{}_j^w\right) + (\boldsymbol{p}^0{}_j^w - \boldsymbol{p}^0{}_r)$$

where $\left(\boldsymbol{p}^0{}_j - \boldsymbol{p}^0{}_j^w\right)$ represents the planar motion and $(\boldsymbol{p}^0{}_j^w - \boldsymbol{p}^0{}_r)$ denotes the parallax motion.

Note that from the third component of 3.5 and the assumptions $\boldsymbol{w}^w{}_j \neq 0$and $\boldsymbol{w}^w \neq 0$. It can be seen that:

$$\frac{\boldsymbol{w}^w{}_j}{\boldsymbol{w}_r} = 1 - \frac{H}{\boldsymbol{w}_r}\frac{\mathrm{e_3}^w}{d_r} = 1 - \gamma \mathrm{e_3}^w \neq 0 \qquad [3.12]$$

And therefore the division of equation 3.9 by $(1-\gamma \mathrm{e_3}^w)$ is valid.

Chapter 4

4 Methods

The camera movement along the optical axis is an important issue for various applications in computer vision. But when the camera is moving forward it might have some rotation and other translations. So we need to cancel out the rotation. There are several methods available to solve this problem. We have found “Plane plus Parallax” method most useful to solve this problem.

4.1 Plane plus Parallax using two frames

Two frames method is comparatively simpler than multi frame method because there is no local phase and global phase calculation in two frames method. As most of the theoretical calculation between two frames and multi frame are same, so we don’t show it separately. We have described the detail theories in chapter 3 for the two frames case and additional local and global phase have been described in chapter 6.2.1 and 6.2.2 for the multi frame case.

We use a number of image frames in multiple frames method and hence this method contains a great deal of independent samples. As a result it provides better output. Moreover, using more frames also increases the signal to noise ratio and further improves the shape reconstruction. However, there are two main benefits for the multi frame estimation which are (i) Overcoming the Aperture problem and (ii) resolving epipolar singularity. Here we have described these two cases in short.

4.1.1 Aperture Problem

When only two images are used as in [6,7], there exist only one epipole. The residual parallax lies along epipolar lines (centered at the epipole, see eq. (1)). The epipolar field provides one line constraint on each parallax displacement, and the brightness constancy constraint forms another line constraint. When those lines are not parallel, their intersection uniquely defines the parallax displacement. However, if the image gradient at an image point is parallel to the epipolar line passing through that point, then its parallax displacement (and hence its structure) cannot be uniquely determined. However, when multiple images with multiple epipoles are used, then this

ambiguity is resolved, because the image gradient at a point can be parallel to at most one of the epipolar lines associated with it. This observation was also made by [12,13].

4.1.2 Epipole Singularity

From the planar parallax equation:

$$\boldsymbol{p}^{0}{}_{j}^{w} - \boldsymbol{p}^{0}{}_{\gamma} = \frac{\gamma}{1-\gamma e_3}(e_3^w \boldsymbol{p}^{0}{}_{\gamma} - \boldsymbol{e}^w)$$

it is clear that the structure γ cannot be determined at the epipole, because at the epipole:

$$(e_3^w \boldsymbol{e} - \boldsymbol{e}^w) = 0$$

and the recovered structure at the vicinity of the epipole is highly sensitive to noise and unreliable. However, when there are multiple epipoles, this ambiguity disappears. The singularity at one epipole is resolved by another epipole.

4.2 Plane plus Parallax using multi frames

Our proposed method is the multi frame method because it is more efficient than two frame method. This algorithm uses more than two frames to recover the structure. Plane registered images should be given as input to the algorithm. Registered images has so many non-linear portions which are not on the plane or parallel to the plane. This method is going to make those portions linear and gives us a much better recovered structure of the reference frame. The outputs of the algorithm are the recovered structure of the reference image and epipoles for each frame with respect to reference frame (in case of forward moving camera epipoles are called focus of expansion (FOE)). This algorithm will be discussed in more details in chapter 6.

Chapter 5

5 Image Motion Estimation

A technique called image registration is used to calculate the relationship between two views of the same planar region in the world. It is possible to find a corresponding pixel of one view in another view of the same surface by using this method. Two important types of image registration for planer region are discussed below. When two images are being registered with respect to planar surface then the planar surface and some parts of the images which are parallel to that plane will have a linear relation. So in registered images there exists non-linear part as well.

5.1 Direct Method Based Image Motion Estimation

Both of the models that used for image motion estimation are discussed below which follow the same framework called hierarchical motion estimation. This framework [4] contains (i) pyramid construction, (ii) motion estimation, (iii) image warping and (iv) coarse-to-fine refinement.

5.1.1 Planar Surface Flow

The rigid motion of a planar surface can be calculated using eight independent parameters [14] also called motion parameters. We are going to describe shortly about this model.

The image motion of a planar surface can be expressed as :

$$u(x) = \frac{1}{Z(x)} A(x)t + B(x)\omega \qquad [5.1.1.1]$$

where Z(x) is the distance of the camera from the point having the image position of (x), then

$$A(x) = \begin{bmatrix} -f & 0 & x \\ 0 & -fy \end{bmatrix} \qquad [5.1.1.2]$$

$$B(x) = \begin{bmatrix} \frac{xy}{f} & -\frac{f+x^2}{f} & y \\ \frac{f+y^2}{f} & -\frac{xy}{f} & -x \end{bmatrix}. \qquad [5.1.1.3]$$

Now from the above, from matrices A and B we can see that they are depending on the image positions and the focal length f which are known and the unknowns parameters are: translation vector t, angular velocity vector ω, and depth Z.

Planar surface equation can be written as

$$k_1X + k_2Y + k_3Z = 1 \qquad [5.1.1.4]$$

where (k_1, k_2, k_3) denoted as surface slant, tilt and the distance of the plane from the origin. Now dividing equation 5.1.1.4 by Z we get

$$\frac{1}{Z} = k_1\frac{x}{f} + k_2\frac{y}{f} + k_3 \qquad [5.1.1.5]$$

Let k to denote the vector (k_1, k_2, k_3) and r to denote the vector $(\frac{x}{f}, \frac{y}{f}, 1)$ than we get

$$\frac{1}{Z(x)} = r(x)^T k \qquad [5.1.1.6]$$

Now replacing the value of 5.1.1.6 in equation 5.1.1.1

$$u(x) = (A(x)t)(r(x)^T k) + B(x)\omega \qquad [5.1.1.7]$$

The flow field can be written as

$$u(x) = a_1 + a_2x + a_3y + a_7x^2 + a_8xy \qquad [5.1.1.8]$$

$$v(x) = a_4 + a_5x + a_6y + a_7xy + a_8y^2 \qquad [5.1.1.9]$$

where 8 parameters $(a_1, \ldots, a_8)$ represent the motion parameters t, ω and the surface parameters(k_1, k_2, k_3).

5.1.2 Rigid Body Model

The rigid motion of a planar surface cannot be solved using a global model; we need to combine this global rigid model with the local model of the surface [15].

As discussed earlier in planar surface model, the rigid body model also has the image motion equation.

$$u(x) = \frac{1}{Z(x)} A(x)t + B(x)\omega \qquad [5.1.2.1]$$

$$A(x) = \begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix} \qquad [5.1.2.2]$$

$$B(x) = \begin{bmatrix} \frac{xy}{f} & -\frac{f + x^2}{f} & y \\ \frac{f + y^2}{f} & -\frac{xy}{f} & -x \end{bmatrix}. \qquad [5.1.2.3]$$

The equation 5.10 relates the global model parameters ω and t with the local model parameters $Z(x)$.

We assume $Z(x)$ is constant over a local image patch. We refine the global and local model parameters using initial estimates. This refinement used to iterate several times.

5.1.3 Plane Registration using Direct Approach

We need two images at a time from image sequence for the registration process. Let's say we have $\emptyset_{j}{}_{j=1}^{l}$ be l images of a rigid scene. Now we choose the reference frame and denote it as $\emptyset_r$ which have the reference plane π that exist in all l images. Now we estimate the camera motion between the images using rigid body motion model and then apply the planar surface model to the selected plane π . These motion parameters used to warp between the reference frame and all other frame which gives a new sequence of images $I_{j}{}_{j=1}^{l}$ where the plane $\pi$is aligned across all frames.

5.2 Feature Based Image Motion Estimation

Feature based image registration is done in two steps. Firstly, a number of control points are selected from the images and correspondence is established between them. Secondly, the position of the corresponding control points between images used to calculate the transformation function which maps the rest of the points in images. Control points can be selected manually or automatically.

We have used imtransform function from image processing toolbox of Matlab to do this registration. Below we will describe the registration process steps

In step 1, we have to read two images. One is input image and another one is base image.

In step 2, we need to select the control points using the tool cpselect-. This tool will enable a graphical user interface which let us to select the control points between images. Input image is the one that going to be warped to be in the coordinate system of the base image.

In step 3, now we can create the TFORM structure using cp2tform function. This function will use the control points obtained from previous steps to infer a spatial transformation or an inverse mapping from output space (*x*, *y*) to input space (*x*, *y*) according to transform type (in our case 'projective'. In a projective transformation, quadrilaterals map to quadrilaterals. Straight lines remain straight. Affine transformations are a subset of projective transformations.). It will return the TFORM structure which contains the spatial transformation.

In step 4, finally we can use the imtransform function to transform the input image to the coordinate system of the base image and the registration is done.

Now in case of plane registration using feature based transformation process similarly we have $\emptyset_j{}_{j=1}^{l}$ images of a rigid scene. The reference plane which we denote by $\emptyset_r$ contains the plane π that is also viewed in all other l images. According to the feature based process we select all control points on the plane π and get the transformation. Using this transformation we can align the plane π between the reference image and all other images. The plane aligned images are denoted as $I_j{}_{j=1}^{l}$. Figure 5.1 shows one image from an image sequence and figure 5.2 is the registered image with respect to reference plane. It is clear from figure 5.2 that the house and features parallel to that become linear.

Figure 5.2.1 : Single image from garden sequence

Figure 5.2.2 : Registered image with respect to front of the house

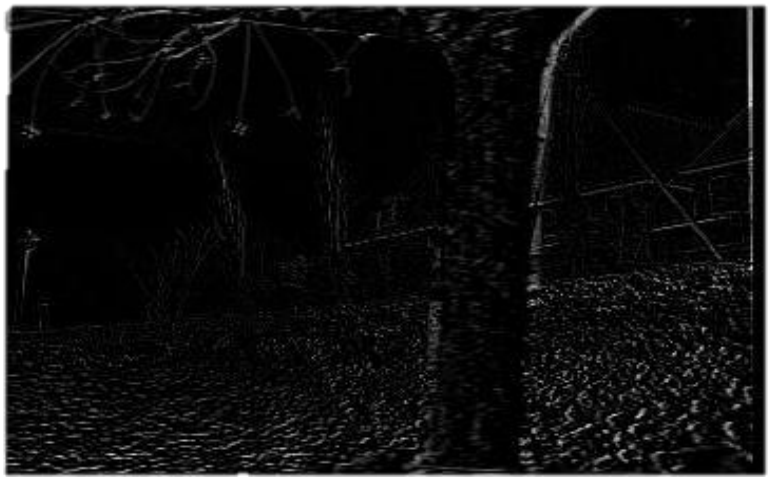

Figure 5.2.3: Deference between images

The non-linear parts of the registered images need to be linear and it will be accomplished by applying “plane+parallax” method on those images that are already aligned with the reference plane.

Chapter 6

6 Multi-frame Planar Parallax Estimation

6.1 Pyramid Construction

Suppose an image is represented initially by the array g_0 which contains C columns and R rows of pixels. Each pixel represents the light intensity at the corresponding image point by an integer, I, between 0 and K-1. This image becomes the bottom, or zero level of the Gaussian pyramid. Pyramid level l contains image g_1, which is a reduced, or low-pass filtered version of g_0 . After that pyramid level 2 contains reduced image g_2 which is half of image g_1. Each level continues in the same way.

The level to level averaging process is performed by the function REDUCE.

g= REDUCE (g_{k-1})

Which means for levels 0 < L < = N

And nodes i, j 0 <= i < C, 0< =j <R1

$$g_L(i,j) = \sum_{m=-2}^{2}.\sum_{n=-2}^{2} \mathrm{w(m,n)}\, g_{L-1}(2i+m, 2j+n) \qquad [6.1.1]$$

Here N refers to the number of levels in the pyramid, while $C_L$and R_L are the dimensions of the L^{th} level.

Note that the density of nodes is reduced by half in one dimension or by a fourth in two dimensions from level to level. The dimensions of the original image are appropriate for pyramid construction if integers M and N exist such that C = $M_c 2^N$+1 and R = $M_R 2^N$+1.

For example if M_c and M_R are both 3 and N is 5 then image measures 97 by 97 pixels. The dimension of g_L are C_L = $M_c 2^{N-1}$+1 and R_L= $M_R 2^{N-1}$+1.

Here w is called the weight or generating kernel which is chosen subject to certain constraints. For simplicity w is made separable as

$$W\ (m,n)\ =\ w(m)\ w(n)$$

On the other hand Laplacian Pyramid is just opposite of Gaussian Pyramid.

6.2 Planar Parallax Motion Estimation

"Plane+Parallax" approach was introduced in [2,5,7,8,16,17]. The concept in that method is, after the alignment of the reference plane, the residual image motion is due only to the translational motion and to the deviations of the scene structure from the planar surface. Plane registration is the process by which the effects of camera rotation and changes in camera calibration are eliminated. The residual image motion (the planar-parallax displacements) forms a radial flow field centered at the epipole. The performance of "plane+parallax" algorithm is better than the traditional camera-centered method which make it an useful framework for 3D shape recovery.

The "Plane+Parallax" framework to recover 3D structure is used in Kumar [16] and Sawhney [7] for two uncalibrated views. Their algorithm solves for structure directly from brightness measurements in two frames and is not applicable for multiple frames.

We have followed the algorithm of Irani [17], which works for multiple frames. There are many problems related to [7, 16] and camera-centered method which are resolved by extending the analysis to multiple frames. In this algorithm we have an image sequence as input which is previously aligned (using [18,1 9]) and the output of the algorithm are the epipoles of all images with respect to reference image, information of 3D structures in the scene relative to a planar surface and the estimation of correspondences of all pixels across all the frames. The information of 3D scene structures and the camera epipoles are computed directly from the image measurements by correcting the errors across the views.

The "Plane+Parallax" algorithm depends on good prior alignment (such as [7, 16]) of the video sequence with respect to a planar surface. That means a large enough real physical surface should be available and visible in all the video sequence. If this type of planar surface does not exist than the algorithm will not work.

The goal is to estimate for each image point $\boldsymbol{p}^0{}_r$ in the reference frame I_r its planar parallax displacement $\boldsymbol{u}^0{}_j^w \stackrel{\text{def}}{=} \boldsymbol{p}^0{}_j^w - \boldsymbol{p}^0{}_r$ between frame I_r *and* I_j.

Assuming an iterative process where $\boldsymbol{u}^0{}_j^{w,prev} = (u^0{}_j^{w,prev} v^0{}_j^{w,prev})^{\boldsymbol{T}}$ is an initial estimate of the parallax image motion, which is given from previous iteration, then:

$$\boldsymbol{u}^0{}_j^w = \boldsymbol{u}^0{}_j^{w,prev} + \boldsymbol{\delta u}^0{}_j^w \qquad [6.2.1]$$

i.e., $\boldsymbol{\delta u}^0{}_j^w = (\boldsymbol{\delta u}^0{}_j^w \boldsymbol{\delta v}^0{}_j^w)^T \stackrel{\text{def}}{=} \boldsymbol{u}^0{}_j^w - \boldsymbol{u}^0{}_j^{w,prev}$. Assuming brightness constancy (namely, that corresponding image points across all frames have a similar brightness value), then:

$$I_r(\boldsymbol{p}^0{}_r) \approx I_j\left(\boldsymbol{p}^0{}_j^w\right) = I_j(\boldsymbol{p}^0{}_r + \boldsymbol{u}^0{}_j^w) = I_j(\boldsymbol{p}^0{}_r + \boldsymbol{u}^0{}_j^{w,prev} + \boldsymbol{\delta u}^0{}_j^w) \qquad [6.2.2]$$

Or:

$$I_r(\boldsymbol{p}^0{}_r - \boldsymbol{\delta u}^0{}_j^w) \approx I_j(\boldsymbol{p}^0{}_r + \boldsymbol{u}^0{}_j^{w,prev}) \qquad [6.2.3]$$

Expanding I_r to its first order taylor series around $\boldsymbol{p}^0{}_r$:

$$I_r(\boldsymbol{p}^0{}_r - \boldsymbol{\delta u}^0{}_j^w) \approx I_r(\boldsymbol{p}^0{}_r) - \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial x}\boldsymbol{\delta u}^0{}_j^w - \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial y}\boldsymbol{\delta v}^0{}_j^w \qquad [6.2.4]$$

From here we get the brightness constraint equation:

$$I_j(\boldsymbol{p}^0{}_r + \boldsymbol{u}^0{}_j^{w,prev}) \approx I_r(\boldsymbol{p}^0{}_r) - \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial x}\boldsymbol{\delta u}^0{}_j^w - \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial y}\boldsymbol{\delta v}^0{}_j^w \qquad [6.2.5]$$

Or:

$$I_j(\boldsymbol{p}^0{}_r + \boldsymbol{u}^0{}_j^{w,prev}) - I_r(\boldsymbol{p}^0{}_r) + \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial x}\boldsymbol{\delta u}^0{}_j^w + \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial y}\boldsymbol{\delta v}^0{}_j^w = 0 \qquad [6.2.6]$$

Substituting $\boldsymbol{\delta u}^0{}_j^w = \boldsymbol{u}^0{}_j^w - \boldsymbol{u}^0{}_j^{w,prev}$ above yields:

$$I_j(\boldsymbol{p}^0{}_r + \boldsymbol{u}^0{}_j^{w,prev}) - I_r(\boldsymbol{p}^0{}_r) + \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial x}(\boldsymbol{u}^0{}_j^w - \boldsymbol{u}^0{}_j^{w,prev}) + \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial y}(\boldsymbol{v}^0{}_j^w - \boldsymbol{v}^0{}_j^{w,prev}) = 0$$

[6.2.7]

or more compactly:

$$I_j^{\tau}(\boldsymbol{p}^0{}_r) + I_x(\boldsymbol{p}^0{}_r)\boldsymbol{u}^0{}_j^w + I_y(\boldsymbol{p}^0{}_r)\boldsymbol{v}^0{}_j^w \approx 0 \qquad [6.2.8]$$

Where

$$I_x(\boldsymbol{p}^0{}_r) \stackrel{\text{def}}{=} \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial x}$$

$$I_y(\boldsymbol{p}^0{}_r) \stackrel{\text{def}}{=} \frac{\partial I_r(\boldsymbol{p}^0{}_r)}{\partial y}$$

$$I_j^{\tau}(\boldsymbol{p}^0{}_r) \stackrel{\text{def}}{=} I_j(\boldsymbol{p}^0{}_r + \boldsymbol{u}^0{}_j^{w,prev}) - I_r(\boldsymbol{p}^0{}_r) - I_x(\boldsymbol{p}^0{}_r)\boldsymbol{u}^0{}_j^{w,prev} - I_y(\boldsymbol{p}^0{}_r)\boldsymbol{u}^0{}_j^{w,prev} \qquad [6.2.9]$$

In the introduction (see eq. 1) we derived an expression for the parallax image motion:

$$\boldsymbol{u}^0{}_j^w \stackrel{\text{def}}{=} \boldsymbol{p}^0{}_j^w - \boldsymbol{p}^0{}_r = \frac{\gamma}{1-\gamma e_3^w}(e_3^w \boldsymbol{p}^0{}_r - e^w) \qquad [6.2.10]$$

This equation is plugged into equation (6.2.9), yielding the epipolar brightness constraint:

$$\left[I_j^{\tau}(\boldsymbol{p}^0{}_r) + I_x(\boldsymbol{p}^0{}_r)\frac{\gamma}{1-\gamma e_3^w}(e_3^w \boldsymbol{x}^0{}_r - e_1^w) + I_y(\boldsymbol{p}^0{}_r)\frac{\gamma}{1-\gamma e_3^w}(e_3^w \boldsymbol{y}^0{}_r - e_2^w)\right] = 0 \qquad [6.2.11]$$

Each pixel and each image frame contributes one such equation, where the unknowns are: the relative scene structure $\gamma = \gamma(\boldsymbol{p}^0{}_r)$ for each pixel $\boldsymbol{p}^0{}_r$, and the epipoles $e^w(j)$ for each frame (j=1,2,…,l, j$\neq r$). Those unknowns are computed in two steps. In the first step, the "Local Phase", the relative scene structure, γ, is estimated by least squares minimization over multiple frames simultaneously, for each pixel $\boldsymbol{p}^0{}_r$ in the reference frame I_r. This phase is followed by the "Global Phase", where all the epipoles $e^w(j)$ are estimated using least squares minimization, between the reference frame I_r and every other frame I_j (j= 1,2,…,l, j$\neq r$). These two phases are described in more details below.

The residual image motion between reference frame and any registered image can be calculated as

$$\boldsymbol{u}^0{}_j^w \stackrel{\text{def}}{=} \boldsymbol{p}^0{}_j^w - \boldsymbol{p}^0{}_r = \frac{\gamma}{1-\gamma e_3^w}(e_3^w \boldsymbol{p}^0{}_r - e^w)$$

Where superscripts j denote the parameters associated with the jth frame, γ is the shape parameter,e_1, e_2 and e_3 are the part of motion parameters e where e= (e_1, e_2, e_3) and t is called the epipole or focusOf expansion. Our main works are to find the epipole and shape parameter from the above equation.

We refine the shape parameter gamma (one value per pixel in the reference frame) and motion parameter t (3 values per frame) in the inner loop of the estimation process. For the very beginning of the process we take the initial value is zero for the shape parameter and [0 0 1] for the motion parameter. Then we refine the value of γ and t in each iteration.

After substituting the above equation of local flow vector we get the following equation directly from image brightness information related to structure and motion parameters.

$$[I_j^{\tau}(\boldsymbol{p}^0{}_r) + I_x(\boldsymbol{p}^0{}_r)\frac{\gamma}{1-\gamma e_3^w}(e_3^w \boldsymbol{x}^0{}_r - e_1^w) + I_y(\boldsymbol{p}^0{}_r)\frac{\gamma}{1-\gamma e_3^w}(e_3^w \boldsymbol{y}^0{}_r - e_2^w)]$$

The equation is referred as the "epipolar brightness constraints". For each pixel and each frame we get such kind of above equation where the unknowns are shape parameter gamma for each pixel and the motion parameter for each frame. We compute these unknowns in two phases which are called local phase and global phase. In the local phase we estimate gamma for each pixel separately via least square minimization over all frames simultaneously. This is followed by global phase where each epipole is estimated between the reference frame and each of other frames using least square minimization over all pixels.

6.2.1 Local Phase

In the local phase we estimate the shape parameter for the reference frame using all the images. Though it is a local quantity but it is common for each image frame. At the very beginning of this phase we assume that all the epipolar parameters are given from the previous iteration and we estimate the unknown scene structure γ from all the images. When epipoles are known (e.g., from the previous iteration), each frame Φ_j provides one constraint of Eq. (3) on γ. However, for increased numerical stability, we locally assume each γ is constant over a small window, Win ($\boldsymbol{p}^0{}_r$), around image point $\boldsymbol{p}^0{}_r$

$$E[\gamma] = \sum_{j} \sum_{(x,y)\in Win(x,y)} \Big(I_\tau(\,1 + \gamma t_3) + \,\gamma \left(I_x(xt_3 - t_1) + I_y(yt_3 - t_2) \right) \Big)^2$$

Where

$$I_\tau = I_j\big(x + u_j, y + v_j\big) - I(x,y) - I_x u_j - I_y v_j$$

From this equation we want to find γ. Theoretically there is sufficient geometric information for solving for γ. However, for increased numerical stability, we locally assume each γ is constant over a small window, Win ($\boldsymbol{p}^0{}_r$), around image point $\boldsymbol{p}^0{}_r$.

$$\text{Win } (\boldsymbol{p}^0{}_r) = [x^0{}_r - h, x^0{}_r + h] \times \,[y^0{}_r - h, y^0{}_r + h]$$

In our experiment we used h=2 (i.e., a 5× 5 $window$).

Therefore for each pixel $\boldsymbol{p}^0{}_r$we minimize the following SSD error (energy) function using all frames:

$$\text{Err}(\gamma) \stackrel{\text{def}}{=} \sum_{j\in\{1,\ldots,l\}\setminus\{r\}} \sum_{\boldsymbol{q}^0\in \text{Win }(\boldsymbol{p}^0{}_r)} \Big(I_j^\tau(\boldsymbol{q}^0)(1 - \gamma e_3^w) + I_x(\boldsymbol{q}^0)\big(e_3^w \boldsymbol{q}^0{}_x - e_1^w\big)\gamma + \big(I_y(\boldsymbol{q}^0)\left(e_3^w \boldsymbol{q}^0{}_y - e_2^w\right)\gamma\Big)2$$

This error term was obtained by multiplying Eq. (7.2.11) by denominator $(1 - \gamma e_3^w)$ to yield a linear equation in γ and equating it to zero yields a single linear equation:

$$\frac{\partial \text{Err}(\gamma)}{\partial \gamma} = 0$$

From which we compute $\gamma(\boldsymbol{p}^0{}_r)$.

For obtaining this goal we have to differentiate it:

For calculation let I_j^τ is denoted as I_τ. After differentiating we get

$$2\Big(I_\tau(\,1 - \gamma e_3) + \,\gamma \left(I_x(xe_3 - e_1) + I_y(ye_3 - e_2) \right) \Big) \big(I_\tau e_3 + \,I_x(xe_3 - e_1) + I_y(ye_3 - e_2) \big) = 0$$

But there is no gamma in $\big(I_\tau e_3 + \,I_x(xe_3 - e_1) + I_y(ye_3 - e_2) \big)$

Let us assume

$$\left(I_\tau e_3 + I_x(xe_3 - e_1) + I_y(ye_3 - e_2)\right) = \text{constant}$$

So we can write

$$\left(I_\tau(1 - \gamma e_3) + \gamma\left(I_x(xe_3 - e_1) + I_y(ye_3 - e_2)\right)\right) = 0$$

$$\Rightarrow\Rightarrow I_\tau - I_\tau\gamma e_3 + \gamma\left(I_x(xe_3 - e_1) + I_y(ye_3 - e_2)\right) = 0$$

$$\Rightarrow\Rightarrow \gamma\left(I_\tau e_3 + I_x(xe_3 - e_1) + I_y(ye_3 - e_2)\right) = I_\tau$$

$$\Rightarrow\Rightarrow \gamma = \frac{I_\tau}{\left(I_\tau e_3 + I_x(xe_3 - e_1) + I_y(ye_3 - e_2)\right)} \quad . \qquad [6.2.1.1]$$

6.2.2 Global Phase

In the global phase we estimate e for every image frame with respect to reference frame. In this phase we assume that γ is given for every pixel from the previous iteration and we estimate for each image I_j the position of its epipole $e_3^w = e_3^w(j) = (e_1^w, e_2^w, e_3^w)^T$ with respect to the reference frame I_r . Multiplying equation (3) by the denominator $(1 - \gamma e_3^w)$ provides a linear equation in the epipole:

$$((\boldsymbol{m}(\boldsymbol{p}^0{}_r))^T \boldsymbol{e}^w = -\mathrm{a}(\boldsymbol{p}^0{}_r) \qquad [6.2.2.1]$$

Where $\boldsymbol{m}(\boldsymbol{p}^0{}_r) = \boldsymbol{m}(\boldsymbol{j}, \boldsymbol{p}^0{}_r) \stackrel{\text{def}}{=} \begin{bmatrix} -I_x(\boldsymbol{p}^0{}_r)\gamma(\boldsymbol{p}^0{}_r) \\ -I_y(\boldsymbol{p}^0{}_r)\gamma(\boldsymbol{p}^0{}_r) \\ \gamma(\boldsymbol{p}^0{}_r)(I_x(\boldsymbol{p}^0{}_r)\boldsymbol{x}^0{}_r + I_y(\boldsymbol{p}^0{}_r)\boldsymbol{y}^0{}_r - I_j^\tau(\boldsymbol{p}^0{}_r)) \end{bmatrix}$

And $\mathrm{a}(\boldsymbol{p}^0{}_r) = \mathrm{a}(j, \boldsymbol{p}^0{}_r) \stackrel{\text{def}}{=} I_j^\tau(\boldsymbol{p}^0{}_r)$.

Note that simple least-squares minimization of the collection of all linear equation (6.2.2.1) (for all image points between the pair of images I_r *and* I_j) is equivalent to a weighted least squares minimization of the collection of equations (6.2.11), with weights $(1 - \gamma e_3^w)$. Because these

weights are not physically meaningful and tend to skew the result, we instead minimized the followed SSD error function:

$$\mathrm{Err}(\boldsymbol{e}^w) \stackrel{\text{def}}{=} \sum_{\boldsymbol{p}^0{}_r \in I_r} (\mathrm{W}\left(\boldsymbol{p}^0{}_r\right)[I_j^\tau\left(\boldsymbol{p}^0{}_r\right)\left(1-\gamma\left(\boldsymbol{p}^0{}_r\right)e_3^w\right)+I_x\left(\boldsymbol{p}^0{}_r\right)(e_3^w\boldsymbol{x}^0{}_r-e_1^w)\gamma(\boldsymbol{p}^0{}_r) + (I_y\left(\boldsymbol{p}^0{}_r\right)\left(e_3^w\boldsymbol{y}^0{}_r-e_2^w\right)\gamma(\boldsymbol{p}^0{}_r)])^2$$

where $\mathrm{W}\left(\boldsymbol{p}^0{}_r\right)$ is a normalized weight: $\mathrm{W}\left(\boldsymbol{p}^0{}_r\right)=\mathrm{W}\left(\boldsymbol{j},\boldsymbol{p}^0{}_r\right)=(1-\gamma\left(\boldsymbol{p}^0{}_r\right)e_3^{w,prev})^{-1}$ and $e_3^{w,prev}=e_3^{w,prev}(j)$ is the third component of the epipole estimated in the previous iteration (and therefore known) between frames I_r *and* I_j.

Let an index i=1,…,n×m run over all the pixels $\boldsymbol{p}^0{}_{i,r}$ of the reference image I_r. Differentiating $\mathrm{Err}(\boldsymbol{e}^w)$ with respect to each coordinate of the epipole$e_3^w=(e_1^w,e_2^w,e_3^w)^T$ and equating it to zero yields a set of linear eqautions in the unknowns (e_1^w,e_2^w,e_3^w):

$$M^TM\boldsymbol{e}^w = -M^T\boldsymbol{b} \qquad [6.2.2.2]$$

Where

$$M = M(j) = \begin{pmatrix} \mathrm{W}\left(\boldsymbol{p}^0{}_{1,r}\right)((\boldsymbol{m}(\boldsymbol{p}^0{}_{1,r}))^T \\ \mathrm{W}\left(\boldsymbol{p}^0{}_{2,r}\right)((\boldsymbol{m}(\boldsymbol{p}^0{}_{2,r}))^T \\ \vdots \\ \mathrm{W}\left(\boldsymbol{p}^0{}_{n\times m,r}\right)((\boldsymbol{m}(\boldsymbol{p}^0{}_{n\times m,r}))^T \end{pmatrix} \qquad [6.2.2.3]$$

And

$$\boldsymbol{b}=\boldsymbol{b}(\boldsymbol{j})=\begin{pmatrix} \mathrm{W}\left(\boldsymbol{p}^0{}_{1,r}\right)(\boldsymbol{a}(\boldsymbol{p}^0{}_{1,r}) \\ \mathrm{W}\left(\boldsymbol{p}^0{}_{2,r}\right)(\boldsymbol{a}(\boldsymbol{p}^0{}_{2,r}) \\ \vdots \\ \mathrm{W}\left(\boldsymbol{p}^0{}_{n\times m,r}\right)(\boldsymbol{a}(\boldsymbol{p}^0{}_{n\times m,r}) \end{pmatrix}. \qquad [6.2.2.4]$$

Therefore:$\boldsymbol{e}^w=-(M^TM)^{-1}M^T\boldsymbol{b}$

We repeat the process for each image I_j, $j \in \{1,2,\dots,l\}\backslash\{r\}$ (one at a time), solving for all epipoles with respect to the reference frame I_r.

We can find $\boldsymbol{e}^w$ by differentiating the following equation with respect to e:

$$\mathrm{Err}(\boldsymbol{e}^w) \stackrel{\text{def}}{=} \sum_{\boldsymbol{p}^0{}_r \in I_r} (\mathrm{W}\left(\boldsymbol{p}^0{}_r\right)[I_j^\tau\left(\boldsymbol{p}^0{}_r\right)\left(1-\gamma\left(\boldsymbol{p}^0{}_r\right)e_3^w\right) + I_x\left(\boldsymbol{p}^0{}_r\right)(e_3^w\boldsymbol{x}^0{}_r - e_1^w)\gamma(\boldsymbol{p}^0{}_r) + (I_y\left(\boldsymbol{p}^0{}_r\right)\left(e_3^w\boldsymbol{y}^0{}_r - e_2^w\right)\gamma(\boldsymbol{p}^0{}_r)])^2$$

$$\frac{\partial \mathrm{Err}(\boldsymbol{e}^w)}{\partial \boldsymbol{e}^w} = 0$$

or

$$\Rightarrow\gg\ 2 * \mathrm{W}\left(\boldsymbol{p}^0{}_r\right)\left(I_\tau(\,1-\gamma e_3) + \gamma\left(I_x(xe_3 - e_1) + I_y(ye_3 - e_2)\right)\right)\mathrm{W}\left(\boldsymbol{p}^0{}_r\right)\left(\gamma I_x + \gamma I_y + (\gamma I_\tau + \gamma x I_x + \gamma y I_y\right) = 0$$

But there is not any term in $\left(\gamma I_x + \gamma I_y + \ (\gamma I_\tau + \gamma x I_x + \gamma y I_y\right)$

Let us assume

$$\left(\gamma I_x + \gamma I_y + \ (\gamma I_\tau + \gamma x I_x + \gamma y I_y\right) = \text{constant}$$

Hence we can write

$$\Rightarrow\gg\ I_\tau(\,1-\gamma e_3) + \gamma\left(I_x(xe_3 - e_1) + I_y(ye_3 - e_2)\right) = 0$$

$$\Rightarrow\gg\ I_\tau - \gamma I_\tau e_3 + \gamma x I_x e_3 - \gamma I_x e_1 + \gamma y I_y - \gamma I_y e_2 = 0$$

$$\Rightarrow\gg\ -I_x e_1 - I_y e_2 + (I_\tau + \ xI_x + yI_y)e_3 = \frac{-I_\tau}{\gamma}$$

$$\Rightarrow\gg\ [-I_x \quad -I_y \quad (I_\tau + \ xI_x + yI_y\,)]\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \frac{-I_\tau}{\gamma}.$$

Let $A = [-I_x \quad -I_x \quad (I_\tau + \ xI_x + yI_y\,)]$, $\mathrm{x} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}$ and $\mathrm{b} = \frac{-I_\tau}{\gamma}$.

So we get the above equation in the form of

$$Ax = b$$

which is very simple to solve to get the value of e .

6.3 Coarse-to-fine Refinement

After iterating a few times at a given resolution level (typically we iterate 5 times at each resolution level) we expand the shape map, γ, so that it can be used as an initial estimate for the structure at the next finer resolution level. We use Gaussian expansion:

Upsample the coarser depth map:

$$\gamma^{fine}(2x^0, 2y^0) = \gamma^{coarse}(x^0, y^0)$$

$$\gamma^{fine}(allotherpixels) = 0$$

(i) Blur with a Gaussian kernel to complete the interpolation:

$$\gamma^{fine} := \frac{4}{16^2}\big((1,4,6,4,1)T(1,4,6,4,1)\big) * \gamma^{fine}$$

Where '*' is the convolution symbol. (The Gaussian kernel is multiplied by 4 to compensate for the sparse input in the first step (i))

Chapter 7

7 Results

In this section we are going to show the results related to real world and synthetic images implementing the algorithm.

7.1 Real World Images

Figure 1 shows pepsi sequence frames in (a,b,c) and the recovered structure in (d). We can see from figure (d) that the proposed algorithm could recover the pepsi can well because it was closer enough to the camera and also having larger motion than any other features.

Figure 7.1.1 : pepsi sequence 1

Figure 7.1.2 : pepsi sequence 2

Figure 7.1.3 : pepsi sequence 3

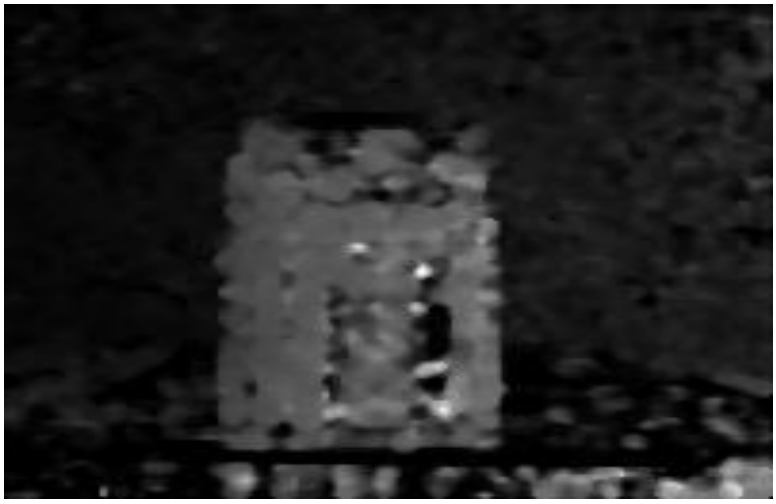

Figure 7.1.4 : Recovered structure of pepsi

Figure 2 shows the garden sequence frames in (a,b,c) and the recovered structure in (d). the recovered structure in (d) shows that the tree has been recovered quite well and also the flowers because of having good texture.

Figure 7.1.5 : Garden sequence 1

Figure 7.1.6 : Garden sequence 2

Figure 7.1.7 : Garden sequence 3

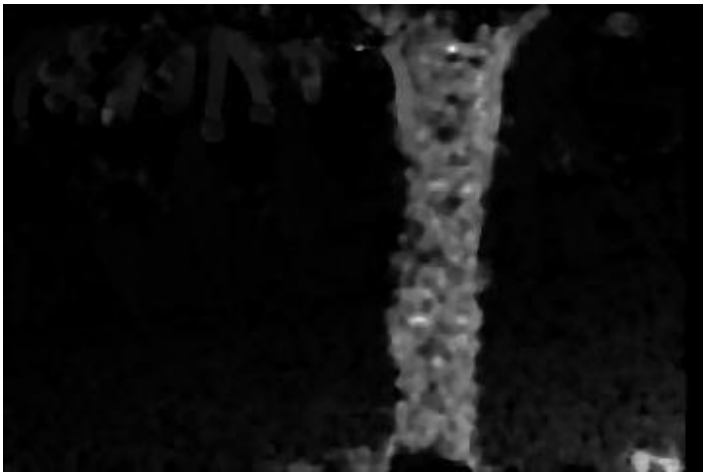

Figure 7.1.8 : Recovered structure of the garden sequence

The results we have got in case of real world images are not satisfactory because the algorithm heavily relies on the plane registration process but our registration process is not good enough. Still we could recover the depth information of 3D structures.

7.2 Synthetic Images

We have shown synthetic images related to three directional motions. Figure 3 shows horizontal motion of the squares (1 pixel in each frame) and recovered structure in (d). Figure 4 shows vertical motions of squares (1 pixel in each frame) and recovered structure in (d). Figure 5 shows both horizontal and vertical motions together and recovered structure in (d).In all cases the background was unmoved which is the same as if the planar surface was aligned in real world images. We can see from the recovered structures that the unmoved background have the dark gray color and the moving squares have light gray color.

Figure 7.2.1 : horizonatal movement 1 pixel

Figure 7.2.2 : horizontal movement 2 pixel

Figure 7.2.3 : horizontal movement 3 pixel

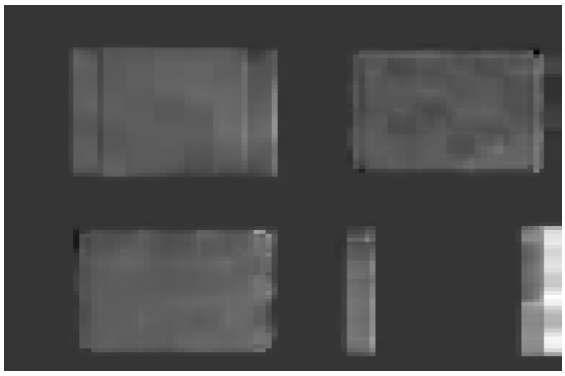

Figure 7.2.4 : recovered horizontal motion

Figure 7.2.5 : vertical movement 1 pixel

Figure 7.2.6 : vertical movement 2 pixel

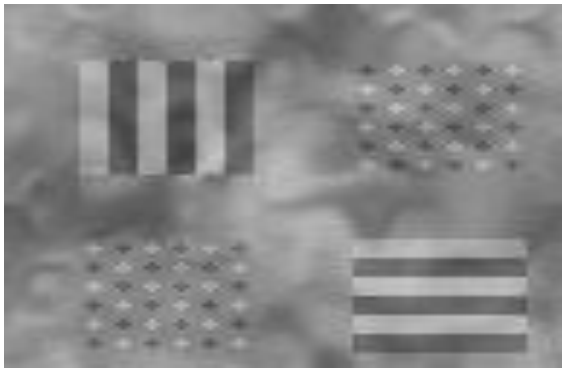

Figure 7.2.7 : vertical movement 3 pixel

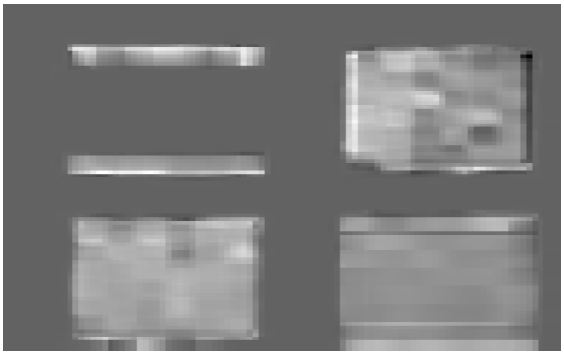

Figure 7.2.8 : recovered vertical motion

Figure 7.2.9 : horizontally moved by 1 pixel

Figure 7.2.10 : vertically moved by 1 pixel

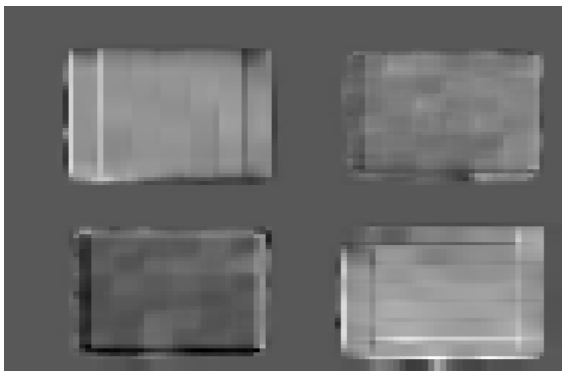

Figure 7.2.11 : recovered structure for mixed motion

Chapter 8

8 Discussion

In this thesis we have studied the forward camera movement to detect 3D structures. To consider the camera movement along only optical axis we have needed to cancel the rotation and other translation. We have used image registration process to cancel the rotation and other translation. All frames have been registered with respect to a reference plane (planar surface). But we could not solve the problem only by the registration process. Because inside the frames there are other regions which does not belongs to the reference plane or parallel to that plane. These regions contain some errors. So we have used "plane+parallax" algorithm for more than two frames to solve the problem.

8.1 Drawback

(i) Firstly, this algorithm gives the privileged role to the reference frame I_r. Information from I_{j1} and I_{j2} is not used explicitly. This cause an asymmetry between the frames.

(ii) The image motion at the coarest level must be less than one pixel due to the taylor series expansion of the brightness assumption. More pyramid levels are needed in case of larger motion. If the texture information is very less in lower frequency than motion recovery in lower resolution level is not good and it will affect the accuracy in the higher resolution level.

(iii) The algorithm is fully dependent on the prior alignment of planar surface. So the quality of the output relies on the quality of the planar surface alignment.

8.2 Future work

The "plane+parallax" algorithm alone is not good enough to solve the problem. We were able to cancel the rotation problem but still the focus of expansion is not on the desired position. In future the frames should be rendered in a way as we obtain the desired result (when it is about focus of expansion). Also for the proposed algorithm future work should come up with the information available from all frames rather than reference frame. Extension of brightness constancy assumption very much needed to increase the usability of this algorithm for more

general case. The planar surface alignment algorithm should work simultaneously with planar parallax algorithm.

References

[1] Torr, P.H.S. and Zisserman, A., *Feature base methods for structure and motion estimation.* In Vision Algorithms: Theory and Practice, pp.278-294, 2000.

[2] Irani M., Rousso B. and peleg P., *Recovery of Ego-Motion Using Region Alignment,* In IEEE Trans. On Pattern Analysis and Machine Intelligence, 19(3), pp.268-272, March 1997.

[3] Adiv G., *Determining Three-Dimensional Motion and Structure from Optical Flow Generated by Several Moving Objects,* In IEEE Trans. On Patterns Analysis and Machine Intelligence , pp.384-401, July 1985.

[4] Bergen J.R., Ananadan P., Hanna K. J. , Hingorani R., *Hierarchical Model-BaseMotion Estimation,* In European Conference on Computer Vision,pages 237- 252,Santa Margarita Ligure, May 1992

[5] Shashua A. and Navab N., *Relative affine structure: Theory and Application to 3D Reconstruction From Perspective Views,* In IEEE Conference on Computer Vision and Pattern Recognition, pages 483-489, 1994B.K.P. Horn. *Robot vision*. MIT press, Cambridge,MA,1986

[6] Kumar R., Ananadan P. and Hanna K., *Direct Recovery of shape From Multiple Views: a parallax Based Approach,* International Conference on Pattern Recognition pp. 685-688, Oct. 1994.

[7] Sawhney H.S., *3D Geometry From Planar Parallax,* In IEEE Conference on Computer Vision and Pattern Recognition , pages 929-934, June 1994.

[8] Irani M. and Anandan P., *Parallax Geometry of Pairs of Points for 3D scene Analysis,* In European Conference on Computer Vision, A, pages 17-30, Cambridge, UK, April 1996.

[9] Irani M., Anandan P., Weinshall D., *From Reference Frames to Reference planes : Multi-View Parallax Geometry and Applications,* In European Conference on Computer Vision, vol.II, pp.829-845,1998.

[10] J.Mundy and A.Zisserman. *Geometric Invariance in Computer Vision*. MIT Press,1992.

[11] Q.-T. Luong and T. Vieville. *Canonic representations for the geometries of multiple projective views*. In Proc. ECCV, pages 589-599, May 1994.

[12] M. Irani , B. Rousso, and S. Peleg. *Computing occluding and transparent motions. International Journal of Computer Vision,* 12:5-16, February 1994.

[13] Faugeras Olivier, *Three Dimensional Computer Vision – A geometric Viewpoint*, MIT Press, Cambridge, MA, 1996.

[14] B.K.P. Horn. *Robot vision*. MIT press, Cambridge,MA,1986

[15] K.J. Hanna. *Direct multi-resolution estimation of ego-motion and structure from motion. In Workshop on Visual Motion*, pages 156-162, Princeton, NJ, October 1991

[16] I.J. Cox, S.L. Hingorani, and S.B. Rao. *A maximum likelihood stereo algorithm. Computer vision and image understanding*, 63(3):542-567, 1996.

[17] M. Irani , P. Anandan. and M. Cohen. *Direct recovery of planar-parallax from multiple frames*. In Vision Algorithms 99, Corfu, September 1999.

[18] Geyer C. , Templeton T. , Meingast M. and Sastry S.S. , *3D Data Processing, Visulaization and Transmission*,In Third International Symposium, 16, pp. 17-24, June 2006.

[19] R. Kumar , P. Anandan, and K. Hanna. *Direct recovery of shape from multiple views: a parallax based approach*. In Proc 12th ICPR, pages 685-688, 1994.