3D Reconstruction of a Scene using Stereo/Multiview Images

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Electronics and Communication Engineering

by

Ayush Kumar Rai
Y11UC073

Tarun Krishna
Y11UC241

Under Guidance of
Mrs. Sonam Nahar

Department of Electronics and Communication Engineering
The LNM Institute of Information Technology, Jaipur

2014-2015

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project entitled 3D Reconstruction of a Scene using Stereo/Multiview Images submitted by Ayush Kumar Rai (Y11UC073) and Tarun Krishna (Y11UC241) in partial fulfillment of the requirement of degree in Bachelor of Technology (B. Tech), is a bonafide record of work carried out by them at the Department of Computer Science and Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during the academic session 2014-2015 under my supervision and guidance and the same has not been submitted elsewhere for award of any other degree. In my opinion, this thesis is of standard required for the award of the degree of Bachelor of Technology (B. Tech).

Date

Supervisor : Mrs. Sonam Nahar

Abstract

Reconstructing a 3-dimensional scene from a set of 2-dimensional images is a fundamental problem in computer vision. A system capable of performing this task can be used in many applications in robotics, architecture, archaeology, biometrics, human computer interaction and the movie and entertainment industry.

This work is basically focused on this most popular area in Computer Vision i.e. Reconstruction of a scene using stereo/multi-view images with known K (calibration) matrix. Initially to develop a deep insight about the epipolar geometry/constraint, we look at the general case of determination of depth through disparity estimation using stereo images. We then survey the stereo correspondence algorithms and compare them qualitatively on the basis of their key properties. Many different approaches have been taken towards solving the stereo correspondence problem and great progress has been made within the field during the last decade. This is mainly thanks to newly evolved global optimization techniques and better ways to compute pixel dissimilarity between views. The most successful algorithms are based on approaches that explicitly model smoothness assumptions made about the physical world, with graph cuts and belief propagation being two frequently used optimisation techniques.

We further extend our discussion to some of the state of the art and top performing procedures regarding stereo/multi-view reconstruction of a scene. We then describe our implementation of this work (including the assumptions that we have made) using the technique of Structure from Motion. This process starts with finding some interesting features using scale invariant SURF feature detector. The features are matched by the help of Brute Force matcher. In order to reject the outliers and estimate the fundamental matrix of the two images, a robust estimation is performed via normalized 8-point algorithms. Two-view reconstruction is finalized by determination of essential matrix from fundamental matrix and further decomposing the essential matrix using single value decomposition and estimating the 3D-point locations as a result of triangulation. The second stage of the reconstruction is the generalization of the two-view algorithm for the N-view case.This goal is accomplished by first reconstructing an initial framework from the first stage and then relating the additional views by finding correspondences between the new view and already reconstructed views. In this way, 3D-2D projection pairs are determined and the camera matrix of this new view is estimated using perspective n point algorithm.

Contents

List of Figures

List of Tables

Chapter *1*

Introduction

1.1 Field of Work

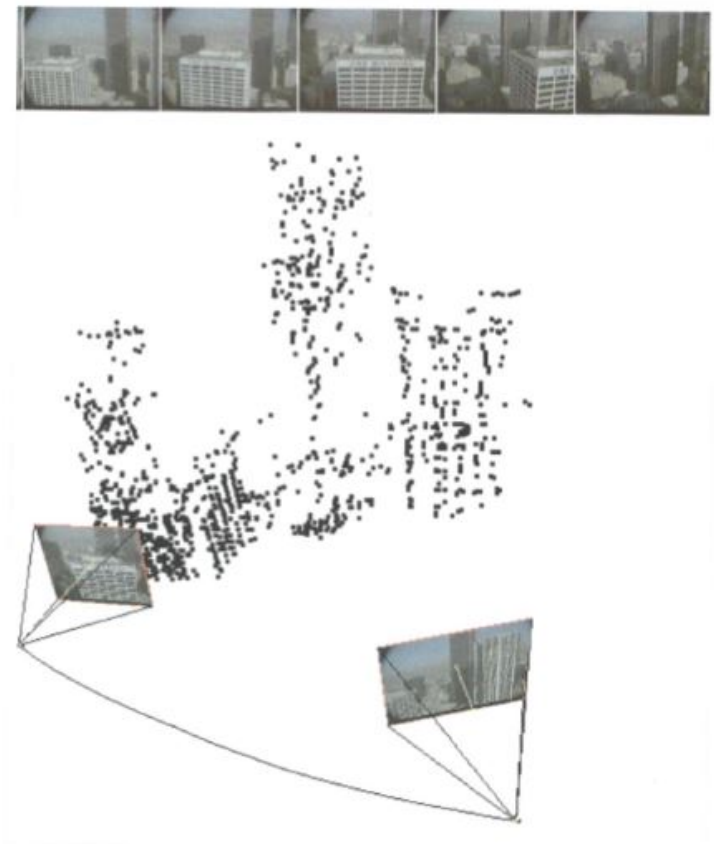

Figure 1.1: Two view

This work addresses one of the most classical problem in the field of Computer Vision i.e. 3D Reconstruction of a scene using stereo/multiview images. During the phenomena of projection of a world coordinate on to the image plane the third dimension or so called depth is lost. In Computer Vision, the main task is to recover the lost third dimension.

Computer vision is a field that includes methods for acquiring, processing, analyzing, and understanding images and, in general, high-dimensional data from the real world in order to produce numerical or symbolic information, e.g., in the forms of decisions. A theme in the development of this field has been to duplicate the abilities of human vision by electronically perceiving and understanding an image.This image understanding can be seen as the determining the symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory. Computer vision has also been described as the enterprise of automating and integrating a wide range of processes and representations for vision perception.As a scientific discipline, computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, or multi-dimensional data from a medical scanner. As a technological discipline, computer vision seeks to apply its theories and models to the construction of computer vision systems.

Stereo/Multi-View stereo consists of reconstructing the three-dimensional (3-D) shape of a scene from a collection of images taken from different vantage points. This is one of the classical problems of computer vision, where significant progress has been made in the last decade. In the early days of stereo, it was common to decompose the problem into two steps: establishing correspondence between points in different views, and then triangulating their position in space. Points in different images are said to be in correspondence when they are images of the same physical point in space via perspective projection. Once correspondence is established, the position of the points as well as the relative pose of the cameras can be determined using well-established procedures.

Unfortunately the first step, establishing correspondence, is far less amenable to a clean and simple solution. First of all, point correspondence can only be reliably established for a very small subset of the scene. For instance, given a scene that contains a white wall, we cannot say which point on one image of the wall corresponds to in another image, since the local appearance is the same for every neighborhood of a point. Therefore, after establishing correspondence and reconstructing the 3-D position of relatively few feature points, one would have to densify the reconstruction by filling in points that cannot be matched from image to image.

1.2 Problem Addressed

In our work, we look at the problem 3D reconstruction of a scene using stereo/multiview images with known K (Calibration) Matrix.We will discuss the idea of Structure from Motion (SfM), or better put as extracting geometric structures from images taken through a camera's motion.First, let us constrain the otherwise lengthy footpath of our approach to using a single camera, usually called a monocular approach, and a discrete and sparse set of frames rather than a continuous video stream. These two constrains will greatly simplify the system we will sketch in the coming pages, and help us understand the fundamentals of any SfM method.

The first discrimination we should make is the difference between stereo (or indeed any multiview), 3D reconstruction using calibrated rigs, and SfM. While a rig of two or more cameras assume we already know what the motion between the cameras is, in SfM we don't actually know this motion and we wish to fi nd it. Calibrated rigs, from a simplistic point of view, allow a much more accurate reconstruction of 3D geometry because there is no error in estimating the distance and rotation between the camerasit is already known.

Let us think for one moment of the goal behind choosing an SfM algorithm. In most cases we wish to obtain the geometry of the scene, for example, where objects are in relation to the camera and what their form is. Assuming we already know the motion between the cameras picturing the same scene, from a reasonably similar point of view, we would now like to reconstruct the geometry. In computer vision jargon this is known as triangulation,

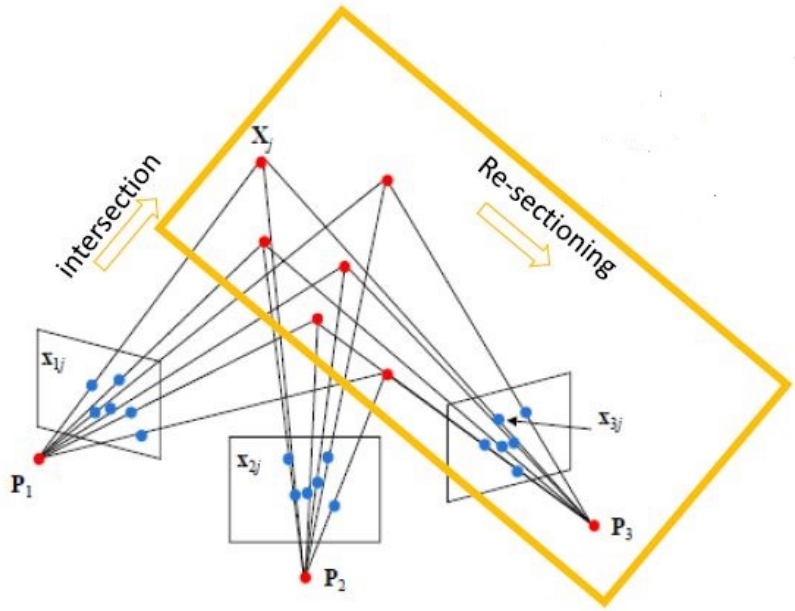

Figure 1.2: Multiview view

and there are plenty of ways to go about it.After we have learned how to recover 3D geometry from two views, we will see how we can incorporate more views of the same scene to get an even richer reconstruction.

Chapter *2*

Literature survey

2.1 Introduction

Computer vision is a vast area of research. Among many interesting areas, the reconstruction of 3-D structure and shape from a number of 2-D images of a scene is still a challenging field for research. 3-D reconstruction is a process to recover the depth and structure of an object or a scene from images. Whenever a scene is captured in an image, the depth information of the scene is lost forever. Practically, there is no way to extract the depth information from a single image of the scene if the objects in the scene have no known surface regularity that can be modeled. There exists a number of methods for depth recovery from multiple images of a scene. Stereo vision, shape from focus, shape from shading and shape from defocus are some examples. Stereo Vision is one of the most popular methods for depth estimation.

The challenges in stereo correspondence estimation make it one of the fundamental subject of investigation in computer vision. There has been numerous research on this subject.Stereo Vision is used primarily for depth estimation and has been in use along with range imaging or range sensors. Range sensors are famous for easy depth estimation but not suitable for all purposes. They are difficult to use with tiny robots. Active sensors affect other devices. They also get affected by other IR and LASER rays. On the other hand, stereoscopy is easier to apply in these sectors. Stereoscopic sensors are immune from spurious readings, cross-talks and multiple reflections present in IR, LASER or SONAR sensors. They are also low cost, realtime, robust and suitable for any size of vehicles. Naturally, considerable amount of research has been done to improve the robustness of stereo algorithms. Currently, researchers try to find better solutions to the challenges. Some of the state of the arts methods are listed with their results in the Middlebury stereo evaluation webpage.A tradeoff exists between accuracy and speed. The methods with higher accuracy suffer from higher implementation and time complexity, whereas, fast and low complexity methods are often of lower accuracy.

3D reconstruction from multiple images is the creation of three-dimensional models from a set of images. It is the reverse process of obtaining 2D images from 3D scenes. The essence of an image is a projection from a 3D scene onto a 2D plane, during which process the depth is lost. The 3D point corresponding to a specific image point is constrained to be on the line of sight. From a single image, it is impossible to determine which point on this line corresponds to the image point. If two images are available, then the position of a 3D point can be found as the intersection of the two projection rays. This process is referred to as triangulation. The key for this process is the relations between multiple views which convey the information that corresponding sets of points must contain some structure and that this structure is related to the poses and the calibration of the camera.

In this chapter we tend to review and describe our detailed and analytical study of problem of depth estimation using stereo images and also reconstruction of a scene using stereo/multiview images.

2.2 Stereo Correspondence Problem

Stereo correspondence has traditionally been, and continues to be, one of the most heavily investigated topics in computer vision. We emphasize calibrated two-frame methods in order to focus our analysis on the essential components of stereo correspondence. A lot of research has been done when it comes to stereo vision. Some surveys have been written to give an overview of popular methods and to compare their performance. Examples of surveys produced within the last decade are [1],[13].A thorough taxonomy of dense stereo correspondence algorithms is [1]. The taxonomy gives a good overview of modern stereo algorithms and their intrinsic components. A continuously updated homepage [17] with a table of performance comparisons between different proposed algorithms comes along the taxonomy. As of today, results from over 80 algorithms are published on the page along with links to papers presenting the algorithms. The homepage also provides several stereo image pairs together with ground truth data. This opens up for the ability to evaluate own algorithms against already existing ones.

Less research has been made for sequences of stereo images than for single stereo image pairs.Some related work that has been found is [7], which describes an algorithm for simultaneously determining optical flow and disparity for stereo sequences by minimizing a global energy function. This algorithm forms the base of [12], which uses the optical flow computed as in the previously mentioned algorithm to make temporally consistent disparity output sequences. Yet another algorithm trying to make use of optical flow is [8], which minimizes an energy function that depends on both disparity and flow. By looking at the outline of these algorithms, they seem rather computationally demanding.

A lot of research in stereo for sequences is carried out by the automotive industry, with detection of vehicles and pedestrians as objective. In [6], the effect of using edge maps for dark scenes (unlitroads) is investigated. Another paper [10] gives an interesting comparison between different disparity estimation techniques for stereo sequences and compares different prediction error metrics that can be used for quality assessment. More information on prediction error metrics for stereo and optical flow is to be found in [20], which discusses some metrics and their benefits and drawbacks. Lastly, [16] presents ways to improve disparity estimates using temporal information and gives some results. One technique that is explored is bilateral filtering of disparity, performed both spatially and temporally.he simplest and popular method for depth recovery is stereo vision or stereo reconstruction that takes a pair of images of a scene and uses a technique commonly called Triangulation to extract the depth from the images. When two image points in the image pairs corresponding to a specific scene point are known, the depth of the scene point can be recovered

Stereo correspondence algorithms search the disparity of the reference image pixels with respect to the target image pixels. There have been numerous researches for accurate disparity estimation. Current state of the arts methods have reached high accuracy but with high computational and implementation complexity. In this chapter, the general workflow of any correspondence estimation method is described. The workflow is followed by a brief description of the basic constraints of stereo vision. The review also describes the broad classification of algorithms followed by the individual descriptions.

In literature, most of the correspondence algorithms follow some basic steps. According to the taxonomy in [1], a general workflow is discussed here. In this step, a dissimilarity measure is employed to compute pixel-wise matching costs. Examples of such measures are Euclidean distance (squared difference), Color difference (absolute or square difference in color space) and Manhattan difference (absolute difference). The computed matching costs for each pixel for all disparities are the outputs of the step. The next step is Cost Aggregation, in this step, the computed matching costs are summed or averaged over a support region in the image space. The support region chosen may be two-dimensional for a fixed disparity to favor fronto-parallel surfaces or three-dimensional in the 3-D space constituted by image dimension (x, y) and disparity values d (xyd space) to favor

slanted surfaces. Two dimensional regions inherently assume the pixels to be from same depth resulting in an unwanted fattening effect. The third step is Disparity Computation and Optimisation. This step is one of the most important step of the workflow that generates the disparity map. In this step, disparity is chosen or computed using the aggregated costs. The type of computation divides the algorithms into two categories - local and global. Local algorithms mainly compute pixel-based disparity and they can simply choose the disparity associated with the minimum cost for each pixel. Global algorithms, on the other hand, optimizes the disparity plane by some energy minimization procedure.The final step refines the disparity map by some postprocessing methods like iterative optimization, curve fitting, mean or median filtering, left-right consistency check or by application of constraints.The output of this step is the final disparity map.

2.3 Multiview Reconstruction of a Scene

The goal of multi-view stereo is to reconstruct a complete 3D object model from a collection of images taken from known camera viewpoints. Over the last few years, a number of high-quality algorithms have been developed, and the state of the art is improving rapidly. In the section [20], a taxonomy of the popular Multiview reconstruction procedures have been explained. Multi-view stereo algorithms can be roughly categorized into four classes.

The first class operates by first computing a cost function on a 3D volume, and then extracting a surface from this volume. A simple example of this approach is the voxel coloring algorithm and its variants [19, 22], which make a single sweep through the volume, computing costs and reconstructing voxels with costs below a threshold in the same pass (note that [2] avoids the need for a threshold). Other algorithms differ in the definition of the cost function and the surface extraction method. A number of methods de- fine a volumetric MRF and use max-flow [15] or multi-way graph cut [11] to extract an optimal surface.The second class of techniques works by iteratively evolving a surface to decrease or minimize a cost function. This class includes methods based on voxels, level sets, and surface meshes. Space carving and its variants progressively remove inconsistent voxels from an initial volume.

Other variants of this approach enable adding as well as deleting voxels to minimize an energy function.Level-set techniques minimize a set of partial differential equations defined on a volume. Like space carving methods, level-set methods typically start from a large initial volume and shrink inward; unlike most space carving methods, however, they can also locally expand if needed to minimize an energy function. Other approaches represent the scene as an evolving mesh [4] that moves as a function of internal and external forces. In the third class are image-space methods that compute a set of depth maps. To ensure a single consistent 3D scene interpretation, these methods enforce consistency constraints between depth maps, or merge the set of depth maps into a 3D scene as a post process [9]. The final class consists of algorithms that first extract and match a set of feature points and then fit a surface to the reconstructed features.

Although most early work in multi-view stereopsis [14] tended to match and reconstruct all scene points independently, recent approaches typically cast this problem as a variational one, where the objective is to find the surface minimizinga global photometric discrepancy functional, regularized by explicit smoothness constraints [1] (a geometric onsistency terms is sometimes added as well). Competingapproaches mostly differ in the type of optimization techniques that they use, ranging from local methods such as gradient descent [3], level sets, or expectation maximization, to global ones such as graph cuts. The variational approach has led to impressive progress, and several of the methods recently surveyed by [4] achieve a relative better accuracy.

However, it typically requires determininga boundingvolume (valid depth range, bounding box, or visual hull) prior to initiatingthe optimization process, which may not be feasible for outdoor scenes and/or cluttered images. propose instead a simple and efficient algorithm for calibrated multi-view stereopsis that does not require any initialization, is capable of detectingand discardingoutliers and obstacles, and outputs a (quasi) dense collection of small oriented rectangular patches [5], obtained from pixel-level correspondences and tightly covering the observed surfaces except in small textureless or occluded regions. It does not perform any smoothingacross nearby features, yet is currently the top performer in terms of both coverage and accuracy for four of the six benchmark datasets provided in [18].

The keys to its performance are effective techniques for enforcinglocal photometric consistency and global visibility constraints. Stereopsis is implemented as a match, expand, and filter procedure, startingfrom a sparse set of matched keypoints, and repeatedly expanding these to nearby pixel correspondences before usingvisibility constraints to filter away false matches. A simple but effective method for turningthe resultingpatch model into a mesh suitable for image-based modeling is also presented. The proposed approach is applied to three classes of datasets: i) objects, where a single, compact object is usually fully visible in a set of uncluttered images taken from all around it, and it is relatively straightforward to extract the apparent contours of the object and compute its visual hull ii) scenes, where the target object(s) may be partially occluded and/or embedded in clutter, and the range of viewpoints may be severely limited, preventingthe computation of effective boundingvolumes (typical examples are outdoor scenes with buildings or walls). iii) crowded scenes, , where moving obstacles appear in different places in multiple images of a static structure of interest (e.g., people passing in front of a building).

Now in the next chapter, we will cover some fundamental concepts of sterevision and multiview geometry, which would help us to develop more insight in multiview geometry.

Chapter *3*

Fundamentals of StereoVision and Multiview Geometry

3.1 The Pinhole Camera Model (Mathematical Model)

The most commonly used model, which we will also use in the course, is the so called pinhole camera. The model is inspired by the simplest cameras. The camera has the shape of a box, light from an object enters though a small hole (the pinhole) in the front and produces and produces an image on the back camera wall.(see figure 3.1)

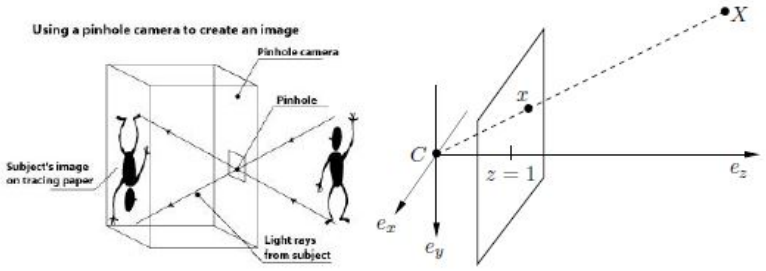

Figure 3.1: The Pinhole camera (left), and a mathematical model (right).

To create a mathematical model we first select a coordinate system (e_x,e_y,e_z).We will refer to this system as the **camera coordinate system**. The origin C = (0, 0, 0) will represent the so called camera center (pinhole). To generate a projection x = (x_1,x_2,1) of a scene point X = (X_1,X_2,X_3) we form the line between X and C and intersect it with the plane z = 1. We will refer to this plane as the image plane and the line as the viewing ray associated with x or X. The plane z = 1 has the normal e_z and lies at the distance 1 from the camera center. We will refer to e_z as the viewing direction. Note that in contrast to a real pinhole camera we have placed the image plane in front of the camera center. This has the effect that the image will not appear upside down as in the real model.

To find the intersection between this line and the image plane z = 1 we need to find and *s* such that the third coordinate sX_3 of *s*X fulfills sX_3 = 1. Therefore, assuming $X_3 \neq 0$, we get *s* = 1/X_3 and the projection.

$$x = \begin{bmatrix} X_1/X_3 \\ X_2/X_3 \\ 1 \end{bmatrix} \tag{3.1}$$

3.1.1 Moving Cameras

In our applications we will frequently have cameras that have been capturing images from different viewpoints. Therefore we need to have a way of modeling camera movements. A camera can undergo translation and rotation. We will represent the translation with a vector $t \in \mathbb{R}^3$ and the rotation with a 3×3 matrix R. Since R is a rotation matrix it has to fulfill $R^T R$ = I and $det(R) = 1$.

To encode camera movements we introduce a new reference coordinate system (e'_x, e'_y, e'_z) , see Figure 3.2. We will refer to this coordinate system as the global coordinate system, since all the camera movements will be related to this system. Typically all the scene point coordinates are also specified in this coordinate system. Lets assume that a scene point X has coordinates (X_1, X_2, X_3) in the camera coordinate system and (X'_1, X'_2, X'_3) in the global coordinate system. Since the camera can be rotated and translated there is a rotation matrix R and translation vector t that relates the two coordinate systems via

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = R \begin{bmatrix} X'_1 \\ X'_2 \\ X'_3 \end{bmatrix} + t \tag{3.2}$$

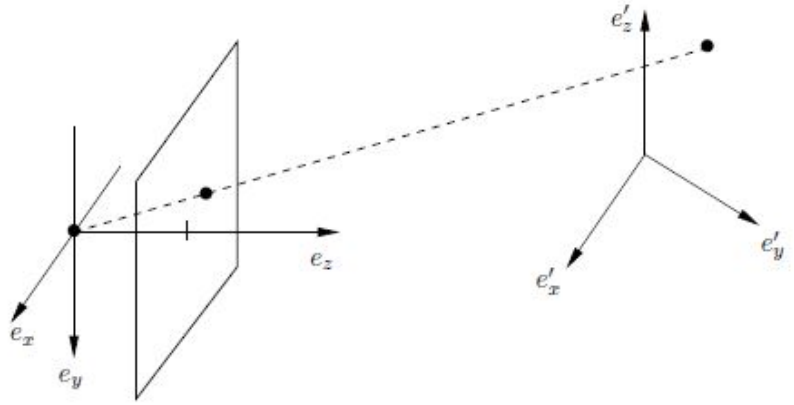

Figure 3.2: New global coordinate system.

If we add an extra 1 to the scene point X we can write 3.2 in matrix form

$$X_3 \begin{bmatrix} X_1/X_3 \\ X_2/X_3 \\ 1 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} X'_1 \\ X'_2 \\ X'_3 \\ 1 \end{bmatrix} \tag{3.3}$$

Here $\begin{bmatrix} R & t \end{bmatrix}$ is the 3×4 matrix where the first 3×3 block is R and the last column is t.As we saw previously, the projection of X is given by 3.1. Therefore we conclude that the projection x of a scene point X (with coordinates given in the global coordinate system) is obtained by computing the vector $v = \begin{bmatrix} R & t \end{bmatrix} = \begin{bmatrix} X \\ 1 \end{bmatrix}$ and dividing the elements of v by the third coordinate of v. From here on we will always assume that scene point coordinates are given in the global coordinate system, if nothing else is stated.

3.1.2 Depth of a Point

We say that a scene point is in front of the camera (or has positive depth) if its third coordinate is positive in the camera coordinate system.

$$X_3 = \begin{bmatrix} R_3 & t_3 \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix} \tag{3.4}$$

where R_3 is the third row R and t_3 is the third coordinate of t. To determine whether a point is in front of the camera we therefore compute $v = \begin{bmatrix} R & t \end{bmatrix} = \begin{bmatrix} X \\ 1 \end{bmatrix}$ and check if the third coordinate is positive.

3.1.3 The Inner Parameters

In the pinhole camera model the image plane is embedded in $\mathbb{R}^3$. That is, image projections are given in the length unit of $\mathbb{R}^3$ (e.g. meters). Furthermore, the center of the image will be located in (0, 0, 1), and will therefore have image coordinate (0, 0). For real cameras we typically obtain images where the coordinates are measured in pixels with (0, 0) in the upper left corner. To be able to do geometrically meaningful computations we need to translate pixel coordinates into the length unit of $\mathbb{R}^3$. We do this by adding a mapping from the image plane embedded in $\mathbb{R}^3$ to the real image, see Figure 3.3.

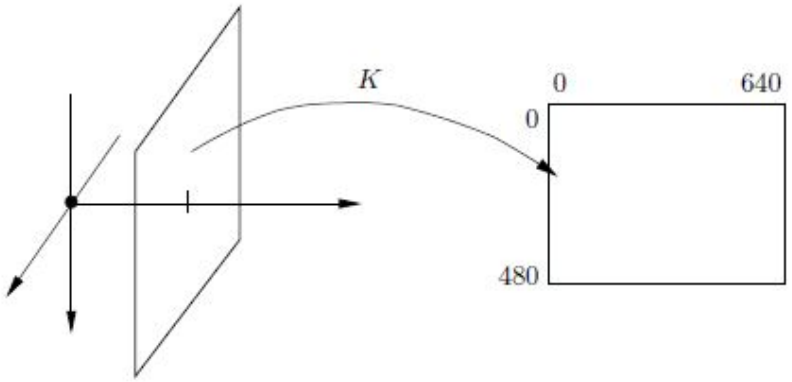

Figure 3.3: The mapping K from the image plane to the real image.

The mapping is represented by an invertible triangular 3×3 matrixK. This matrix contains what is usually referred to as the inner parameters of the camera, that is, focal length, principal point etc. The projection (reference

coordinate system to real image) is now given by :

$$\lambda \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = K \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} X_1' \\ X_2' \\ X_3' \\ 1 \end{bmatrix} \tag{3.5}$$

or in matricx form

$$\lambda \mathbf{x} = P\mathbf{X} \tag{3.6}$$

The matrix K is an upper triangular matrix with the following shape:

$$K = \begin{bmatrix} \gamma f & sf & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

The parameter f is called the **focal length**. This parameter re-scales the image coordinates into pixels. The point $(x_0; y_0)$ is called the principal point. For many cameras it is enough to use the focal length and principal point. In this case the K matrix transforms the image points according to :

$$\begin{bmatrix} fx + x_0 \\ fy + y_0 \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.8}$$

3.2 Epipolar Geometry and Fundamental Matrix

The epipolar geometry is the intrinsic projective geometry between two views. It is independent of scene structure, and only depends on the cameras internal parameters and relative pose.

The fundamental matrix F encapsulates this intrinsic geometry. It is a 3×3 matrix of rank 2. If a point in 3-Dspace X is imaged as x in the first view, and x' in the second, then the image points satisfy the relation $x'^T F x = 0$.

3.2.1 Epipolar Geometry

The epipolar geometry between two views is essentially the geometry of the intersection of the image planes with the pencil of planes having the baseline as axis (the baseline is the line joining the camera centres). This geometry is usually motivated by considering the search for corresponding points in stereo matching, and we will start from that objective here.

Suppose a point X in 3-space is imaged in two views, at x in the first, and x' in the second. What is the relation between the corresponding image points x and x'? As shown in Figure3.4 the image points x and x', space point X, and camera centres are coplanar. Denote this plane as π. Clearly, the rays back-projected from x and x' intersect at X, and the rays are coplanar, lying in π. It is this latter property that is of most significance in searching for a correspondence.

Supposing now that we know only x, we may ask how the corresponding point x' is constrained. The plane π is determined by the baseline and the ray defined by x. From above we know that the ray corresponding to the

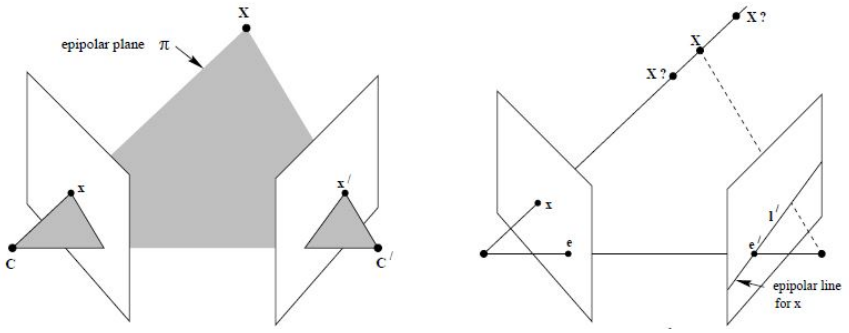

Figure 3.4: **Point correspondence geometry**. (a) The two cameras are indicated by their centres **C** and **C** and image planes.The camera centres, 3-space point X, and its images x and x' lie in a common plane π. (b) An image point x back-projects to a ray in 3-space defined by the first camera centre, C, and x. This ray is imaged as a line l' in the second view. The 3-space point X which projects to x must lie on this ray, so the image of X in the second view must lie on l'.

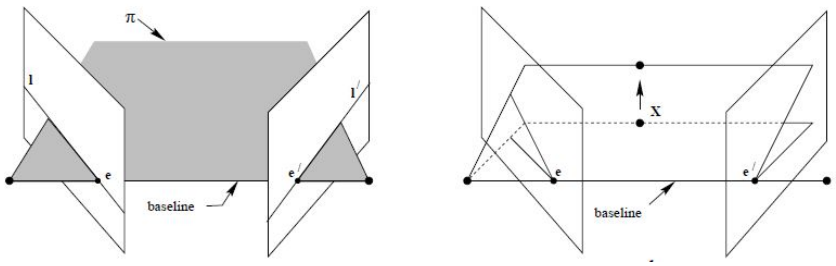

Figure 3.5: **Epipolar geometry**. (a) The camera baseline intersects each image plane at the epipoles e and e'. Any plane π containing the baseline is an epipolar plane, and intersects the image planes in corresponding epipolar lines l and l'. (b) As the position of the 3D point X varies, the epipolar planes rotate about the baseline. This family of planes is known as an epipolar pencil. All epipolar lines intersect at the epipole.

(unknown) point x' lies in π , hence the point x' lies on the line of intersection l' of π with the second image plane. This line l' is the image in the second view of the ray back-projected from x. It is the epipolar line corresponding to x. In terms of a stereo correspondence algorithm the benefit is that the search for the point corresponding to x need not cover the entire image plane but can be restricted to the line l'.

3.2.2 The Fundamental Matrix

The fundamental matrix is the algebraic representation of epipolar geometry. In the following we derive the fundamental matrix from the mapping between a point and its epipolar line, and then specify the properties of the matrix.

Given a pair of images, it was seen in Figure 3.4 that to each point x in one image, there exists a corresponding epipolar line l' in the other image.

Any point x' in the second image matching the point x must lie on the epipolar line l'. The epipolar lineis the projection in the second image of the ray from the point x through the camera centre C of the first camera. Thus, there is a map $x \mapsto l'$from a point in one image to its corresponding epipolar line in the other image. It is the nature of this map that will now be explored. It will turn out that this mapping is a (singular) correlation, that is a projective mapping from points to lines, which is represented by a matrix F, the fundamental matrix.

Thus we can say that the fundamental matrix satisfies the condition that for any pair of corresponding points $x \leftrightarrow x'$ in the two images $x'^T F x = 0$.This is true, because if points x and x' correspond, then x' lies on the epipolar line $l' = Fx$ corresponding to the point x. In other words $0 = x'^T l' = x'^T F x$. Conversely, if image points satisfy the relation $x'^T F x = 0$ then the rays defined by these points are coplanar. This is a necessary condition for points to correspond.

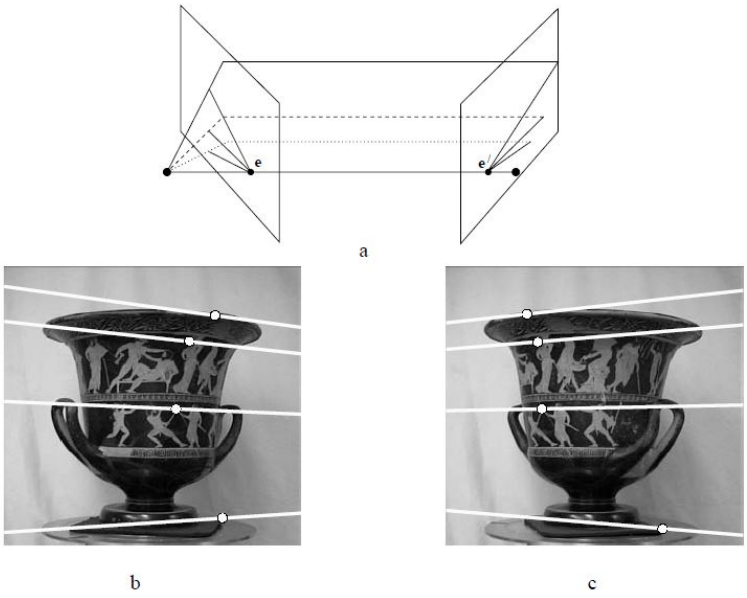

Figure 3.6: **Motion parallel to the image plane**.In the case of a special motion where the translation is parallel to the image plane, and the rotation axis is perpendicular to the image plane, the intersection of the baseline with the image plane is at infinity. Consequently the epipoles are at infinity, and epipolar lines are parallel. (a) Epipolar geometry for motion parallel to the image plane. (b) and (c) a pair of images for which the motion between views is (approximately) a translation parallel to the x-axis, with no rotation. Four corresponding epipolar lines are superimposed in white. Note that correspondingpoints lie on corresponding epipolar lines.

3.2.3 Computation of the Fundamental Matrix F

The fundamental matrix is defined by the equation asdefined in the previous section :

$$x'^T F x = 0 \tag{3.9}$$

for give any pair of matching points $x \leftrightarrow x'$ in two images. Given sufficiently many point matches $x_i \leftrightarrow x'_i$ (at least 7), equation (3.9) can be used to compute the unknown matrix F. In particular, writing $x = (x, y, 1)^T$ and $x' = (x', y', 1)^T$ each point match gives rise to one linear equation in the unknown entries of F. The coefficients of this equation are easily written in terms of the known coordinates x and x'. Specifically, the equation corresponding to a pair of points (x, y, 1) and (x', y', 1) is

$$x'x f_{11} + x'y f_{12} + x' f_{13} + y'x f_{21} + y'y f_{22} + y" f_{23} + x f_{31} + y f_{32} + f_{33} = 0 \tag{3.10}$$

Denote by **f** the 9-vector made up of the entries of F in row-major order. Then (3.10) can be expressed as a vector inner product

$$(x'x, x'y, x', y'x, y'y, y', x, y, 1)f = 0$$

.

From a set of n point matches, we obtain a set of linear equations of the form

$$Af = \begin{bmatrix} (x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} f = 0 \tag{3.11}$$

This is a homogeneous set of equations, and f can only be determined up to scale. For a solution to exist, matrix A must have rank at most 8, and if the rank is exactly 8, then the solution is unique (up to scale), and can be found by linear methods the solution is the generator of the right null-space of A.

If the data is not exact, because of noise in the point coordinates, then the rank of A may be greater than 8 (in fact equal to 9, since A has 9 columns). In this case, one finds a least-squares solution.

3.3 The Essential Matrix

The essential matrix is the specialization of the fundamental matrix to the case of normalized image coordinates (see below). Historically, the essential matrix was introduced (by Longuet-Higgins) before the fundamental matrix, and the fundamental matrix may be thought of as the generalization of the essential matrix in which the (inessential) assumption of calibrated cameras is removed. The essential matrix has fewer degrees of freedom, and additional properties, compared to the fundamental matrix. These properties are described below.

Normalized coordinates. Consider a camera matrix decomposed as $P = K[R|t]$, and let $x = PX$ be a point in the image. If the calibration matrix **K** is known, then we may apply its inverse to the point x to obtain the point $\hat{x} = K^{-1}x$. Then $\hat{x} = [R|t]X$, where $\hat{x}$ is the image point expressed in normalized coordinates. It may be thought of as the image of the point X with respect to a camera $[R|t]$ having the identity matrix I as calibration matrix. The camera matri $xK^{-1}P = [R|t]$ is called a **normalized camera matrix**, the effect of the known calibration matrix having been removed.

Now, consider a pair of normalized camera matrices$P = [I|0]$ and $P' = [R|t]$. The fundamental matrix corresponding to the pair of normalized cameras is customarily called the essential matrix

$$E = [t]_\times R = R[R^T t]_\times. \tag{3.12}$$

The defining equation for the essential matrix is

$$\hat{x}'^T E\hat{x} = 0 \tag{3.13}$$

in terms of the normalized image coordinates for corresponding points $x \leftrightarrow x'$. Substituting for $\hat{x}$ and $\hat{x}'$ gives $x'^T K^{-T} E K^{-1} x = 0$. Comparing this with the relation $x'^T Fx = 0$ for the fundamental matrix, it follows that the relationship between the fundamental and essential matrices is

$$E = \hat{K}^T FK \tag{3.14}$$

3.3.1 Extraction of cameras from the essential matrix

Once we have determined the essential matrix E we need extract cameras from it. Basically we want to decompose it into $E = SR$ where S is a skew symmetric matrix and R is a rotation. We will use the two matrices

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} and \quad Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.15}$$

The matrix W is a rotation and Z is skew symmetric. Furthermore, for these matrices we have that

$$ZW = diag([110]) \tag{3.16}$$

$$ZW^T = -diag([110]) \tag{3.17}$$

If E has the **SVD** then we can now find two solutions;$E = S_1R_1$, where

$$S_1 = -UZU^T, R_1 = UW^TV^T \tag{3.18}$$

and $E = S_2R_1$, where

$$S_2 = UZU^T, R_2 = UWV^T : \tag{3.19}$$

To see that these are valid solutions we first verify that R_1 and R_2 are rotations. Since

$$R_1^TR_1 = (UW^TV^T)^TUW^TV^T = VWU^TUW^TV^T = I \tag{3.20}$$

R_1 is orthogonal. Furthermore,

$$det(R_1) = det(UW^TV^T) = det(U)det(W^T)det(V^T) = det(W)det(UV^T) = 1; \tag{3.21}$$

and therefore R_1 is a rotation. (Note that if $det(UV^T) = -1$ then theR_1 that we obtain is not a rotation but a rotation composed with a reflexion and therefore not a valid solution.) ThatS_1 is skew symmetric is easy to see since

$$-S_1T = (UZU^T)^T = UZ^TU^T = -UZU^T = S_1 : \tag{3.22}$$

Finally we see that

$$S_1R_1 = -UZU^TUWV^T = -UZW^TV^T = -U(-diag([110]))V^T = E \tag{3.23}$$

and therefore S_1R_1 is a valid decomposition.$E = S_2R_2$ can be verified similarly. When we have determined a decomposition E = SR we need to compute a translation vector t from S such that $[t]_x = S$. For such a t we have

$$St = [t]_xt = t \times t = 0 : \tag{3.24}$$

Therefore the vector t is in the null space of S. The null space of the two matrices S1 and S2 are the same and we can find it by looking in the third column of U. Note that if t is in the null space of S then so is λt. In fact any non-zero λ gives a valid solution since $[\lambda t]_xR = [\lambda t]_xR = \lambda E$ which is also a valid essential matrix for the problem. Different λ corresponds to rescaling the solution and since there is a scale ambiguity we cannot

determine a "true" value of λ. However the sign ofλ is important since it determines whether points are in front of the cameras or not in the final reconstruction. To make sure that we can find a solution where the points are in front of both the cameras we therefore test $\lambda = -1$.

Ifu_3 is the third column of U we get the four solutions

$$P2 = [UWV^T u_3] or [UW^T V^T u_3] \quad (from \lambda = 1)) \tag{3.25}$$

$$P2 = [UWV^T - u_3] or [UW^T V^T - u_3] \quad (from \lambda = -1) \tag{3.26}$$

When we have computed these four solutions we compute the 3D points using triangulation for all the choices of P_2 and select the one with where points are in front of both P_1 and P_2.

3.4 Finding P: The Resection Problem

In this section we will outline a method for finding the camera matrix P. We are assuming that the scene points X_i and their projections x_i are known. The goal is to solve the equations :

$$\lambda_i x_i = PX, i = 1....N \tag{3.27}$$

where the λ_i and P are the unknowns. This problem, determining the camera matrix from know scene points and projections is called the resection problem. The 3×4 matrix P has 12 elements but the scale is arbitrary and therefore 11 degrees of freedom. There are 3N equations (3 for each point projection), but each new projection introduces one additional unknown λ'. Therefore we need

$$3N \geq 11 + N \Rightarrow N \geq 6 \tag{3.28}$$

points in order for the problem to be well defined. To solve the problem we will use a simple approach called Direct Linear Transformation (DLT). This method formulates a homogeneous linear system of equations and solves this by finding an approximate null space of the system matrix. If we let p_i,$i = 1, 2, 3$ be 4×1 vectors containing the rows of PS, that is,

$$P \begin{bmatrix} p_1^T \\ p_2^T \\ p_3^T \end{bmatrix} \tag{3.29}$$

then we can write 3,12 as

$$X_i^T p_1 - \lambda_i x_i = 0 \tag{3.30}$$

$$X_i^T p_2 - \lambda_i y_i = 0 \tag{3.31}$$

$$X_i^T p_3 - \lambda_i = 0 \tag{3.32}$$

where $x_i = (x_i, y_i, 1.$ In matrix form this can be written

$$\begin{bmatrix} X_i^T & 0 & 0 & -x_i \\ 0 & X_i^T & 0 & -y_i \\ 0 & 0 & X_i^T & -1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \lambda_i \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.33}$$

Note that since X_i is a 4×1 vector each 0 on the left hand side actually represents a 1×4 block of zeros. Thus the left hand side is a 3×13 matrix multiplied with a 13×1 vector. If we include all the projection equations in one matrix we get a system of the form

$$MP = \begin{bmatrix} X_1^T & 0 & 0 & -x_1 & 0 & 0 & 0 & \dots & \dots \\ 0 & X_1^T & 0 & -y_1 & 0 & 0 & 0 & \dots & \dots \\ 0 & 0 & X_1^T & -1 & 0 & 0 & 0 & \dots & \dots \\ X_2^T & 0T & 0 & 0 & -x_2 & 0 & 0 & \dots & \dots \\ 0 & X_2^T & 0 & 0 & 0 & -y_2 & 0 & \dots & \dots \\ 0 & 0 & X_2^T & 0 & 0 & 0 & -1 & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \vdots \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ \vdots \\ \vdots \end{bmatrix} \tag{3.34}$$

Here we are interested in finding a non-zero vector in the nullspace of **M** which can be solved using **Least Square Method**.

3.5 Linear Triangulation

The linear triangulation method is the direct analogue of the DLT method . In each image we have a measurement $x = PX$, $x' = P"X$, and these equations can be combined into a form $AX = 0$, which is an equation linear in X.

First the homogeneous scale factor is eliminated by a cross product to give three equations for each image point, of which two are linearly independent. For example for the first image, $x \times (PX) = 0$ and writing this out gives

$$x(p_3^T X) - (p_1^T X) = 0 \tag{3.35}$$

$$y(p_3^T X) - (p_2^T X) = 0 \tag{3.36}$$

$$x(p_2^T X) - y(p_1^T X) = 0 \tag{3.37}$$

where p_i^T are the rows of P. These equations are linear in the components of X. An equation of the form AX = 0 can then be composed, with

$$A = \begin{bmatrix} xp_3^T X & - & p_1^T X = 0 \\ yp_3^T X & - & p_2^T X = 0 \\ x'(p_3')^T X & - & (p_1')^T X = 0 \\ y'(p_3')^T X & - & (p_2')^T X = 0 \end{bmatrix} \tag{3.38}$$

where two equations have been included from each image, giving a total of four equations in four homogeneous unknowns. By setting$X = (X, Y, Z, 1)^T$ the set of homogeneous equations, AX = 0, is reduced to a set of four inhomogeneous equations in three unknowns. The least-squares solution to these inhomogeneous equations can be used to get the solutions

Chapter *4*

Proposed Work

4.1 Introduction

In this chapter. we clearly describe our implementation of stereo/multiview reconstruction of a scene and the assumptions that we have taken into account while tackling this problem. From [21], it becomes quite evident that the problem of stereo/multiview reconstruction can be worked out using three categories of datasets i.e. from uncalibrated raw images, images with known internal parameters and finally from images with all parameters (trivial case). In our implementation, we adopt the second approach (images with known Calibration Matrix). We extensively use Fountain P-11 datasets, herz-jesu P-8 and castle P-19 datasets available at the EPFL website to experiment with our procedures.

There are a number of assumptions that we have taken into consideration in order to make our task relatively simpler and easy to understand, implement and analyse. We assume that all the images are taken by one hardware(camera) only, all images are rectified, internal camera parameters are known to us, the images in the dataset are not the consecutive frames of a video, all objects are lambertian and lastly the scene is stationary and every time it is camera that is in motion.Lastly in fig 4.6, we have explained our adopted approach in the form of a flow-chart.

4.2 Estimating cameras and reconstructing the sparse geometry

Our approach to reconstruct 3D geometry from a collection of photos , requires accurate information about the relative location,orientation, and intrinsic parameters such as focal length for each photograph in a collection.Many digital cameras embed focal length and other information in the exif tags of image files. These values are useful for initialization, but are sometimes inaccurate. In our system, we do not rely on the camera or any other piece of equipment to provide us with location, orientation, or geometry. Instead, we compute this information from the images themselves using computer vision techniques such as Structure from Motion. We first detect feature points in each image, then match feature points between pairs of images, and finally run an iterative, robust Structure from Motion procedure to recover the camera parameters.Throughout our work we have assumed the use of a calibrated camera one that was calibrated beforehand. Calibration is a ubiquitous operation in computer vision. We therefore assume the existence of the camera's intrinsic parameters embodied in the K matrix, one of the outputs from the calibration process.

4.3 Feature detection and matching

The first step is to find feature points in each image. We use the SURF(Speeded Up Robust Features) keypoint detector, because this technique ensures that the points of interest are scale invariant.A typical image contains several thousand SURF keypoints. Other feature detectors could also potentially be used like SIFT keypoint detector and descriptor.In addition to the keypoint locations themselves, SURF provides a local descriptor for each keypoint.Next, for each pair of images, we match keypoint descriptors between the pair, using the BFmatcher, which is the most straightforward way to match two feature sets implemented by comparing each feature in the first set to each feature in the second set (hence the phrasing brute-force) and getting the best match.Practically, raw matching like we performed initially is good only up to a certain level, and many matches are probably erroneous. For that reason, most SfM methods perform some form of filtering on the matches to ensure correctness and reduce errors. One form of filtering, which is the built-in OpenCV's brute-force matcher, is cross-check filtering. That is, a match is considered true if a feature of the first image matched a feature of the second image, and the reverse check also matched the feature of the second image with the feature of the first image. Another common filtering mechanism, which we used is to filter based on the fact that the two images are of the same scene and have a certain stereo-view relationship between them i.e we robustly estimate a fundamental matrix for the pair using RANSAC. During each RANSAC iteration, we compute a candidate fundamental matrix using the eight-point algorithm [Hartley and Zisserman 2004]. Finally, we remove matches that are outliers to the recovered fundamental matrix. If the number of remaining matches is less than twenty(in our case), we remove all of the matches from consideration.After finding a set of geometrically consistent matches between each image pair, we organise them in the order of higest matches between each pair.

4.4 Structure from motion

Next, we recover a set of camera parameters and a 3D location for each pair. The recovered parameters should be consistent, in that the reprojection error should be less.SfM problems are particularly prone to getting stuck in bad local minima, so it is important to provide good initial estimates of the parameters. Rather than estimating the parameters for all cameras and pairs at once, we take an incremental approach, adding in one camera at a time.We begin by estimating the parameters of a single pair of cameras. This initial pair should have a large number of matches, but also have a large baseline, so that the 3D locations of the observed points are well-conditioned. We therefore choose the pair of images that has the largest number of matches.

4.4.1 Finding camera matrices(two view geometry)

Now that we have obtained matches between keypoints, we can calculate the fundamental matrix and from that obtain the essential matrix. However, we must first align our matching points into two arrays, where an index in one array corresponds to the same index in the other. This is done to fulfill the epipolar constraints which is required for the calculation of fundamental matrix.Once we have calculated Fundamental matrix we can easily calculate Essential matrix which is simply:

$$E = K'.F.K \tag{4.1}$$

Now we are ready to find the camera matrices.We are going to use a very straightforward and simplistic implementation of it, and OpenCV makes things very easy for us. But first, we will briefly examine the structure of the camera matrix we are going to use.

(a) Left Image

(b) Right Image

Figure 4.1: Two Images Under Consideration for Matching

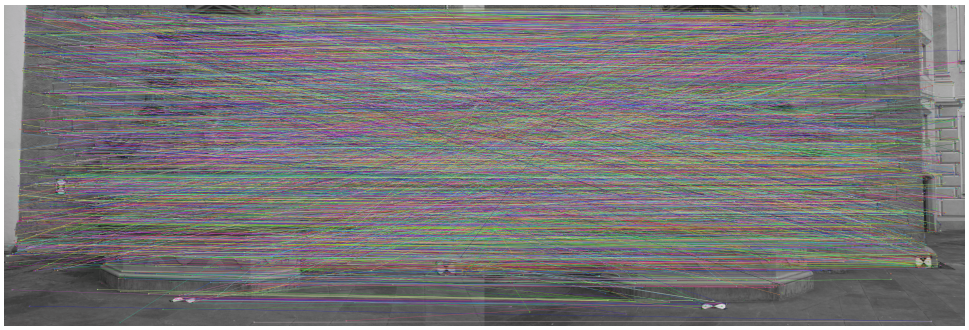

Figure 4.2: Initial Matches

Figure 4.3: Matches after removing outliers

$$P = [R|t] = \begin{bmatrix} r1 & r2 & r3 & t1 \\ r4 & r5 & r6 & t2 \\ r7 & r8 & r9 & t3 \end{bmatrix} \tag{4.2}$$

This is the model for our camera, it consists of two elements, rotation (denoted as R) and translation (denoted as t). The interesting thing about it is that it holds a very essential equation: **x=PX**, where **x** is a 2D point on the image and **X** is a 3D point in space. There is more to it, but this matrix gives us a very important relationship between the image points and the scene points. All we had to do is take the Singular Value Decomposition **(SVD)** of the essential matrix we obtained from before, and multiply it by a special matrix **W**.The SVD operation decomposed our matrix E into two parts, a rotation element and a translation element. In fact, the essential matrix was originally composed by the multiplication of these two elements as discussed in previous chapters.

One thing to notice what we just did only gives us one camera matrix, so where is the other camera matrix? Well, we perform this operation under the assumption that one camera matrix is fixed and canonical (no rotation and no translation). The next camera matrix is also canonical:

$$P_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.3}$$

The other camera that we recovered from the essential matrix has moved and rotated in relation to the fixed one. This also means that any of the 3D points that we recover from these two camera matrices will have the

first camera at the world origin point (0, 0, 0). Many a times the calculation of the fundamental matrix from the point matching is erroneous, and this affects the camera matrices. Continuing triangulation with faulty camera matrices is pointless. We can install a check to see if the rotation element is a valid rotation matrix. Keeping in mind that rotation matrices must have a determinant of 1 (or -1). At this point we have the two cameras that we need in order to reconstruct the scene. The canonical first camera, in the P_0 variable, and the second camera we calculated, from the fundamental matrix in the P_1 variable. The next section will reveal how we use these cameras to obtain a 3D structure of the scene.

4.4.2 Reconstructing the scene

Now we perform Triangulation.In the preceding section we obtained two camera matrices from the essential and fundamental matrices; we already discussed how these tools will be useful for obtaining the 3D position of a point in space. Remember we had two key equations arising from the 2D point matching and P matrices: x=PX and x'=P'X, where x and x' are matching 2D points and X is a real world 3D point imaged by the two cameras. If we rewrite the equations, we can formulate a system of linear equations that can be solved for the value of X, which is what we desire to find. Assuming X = (x, y, z, 1)t (a reasonable assumption for points that are not too close or too far from the camera center) creates linear equation system of the form AX = B.

This will give us an approximation for the 3D points arising from the two 2D points. One more thing to note is that the 2D points are represented in homogenous coordinates, meaning the x and y values are appended with a 1.We should make sure these points are in normalized coordinates, meaning that they were multiplied by the calibration matrix K beforehand. We may notice that instead of multiplying each point by the matrix K we can simply make use of the KP matrix (the K matrix multiplied by the P matrix)

In the following image we will see a triangulation result of two images . The two images at the top are the original two views of the scene, and the bottom pair is the view of the reconstructed point cloud from the two views.

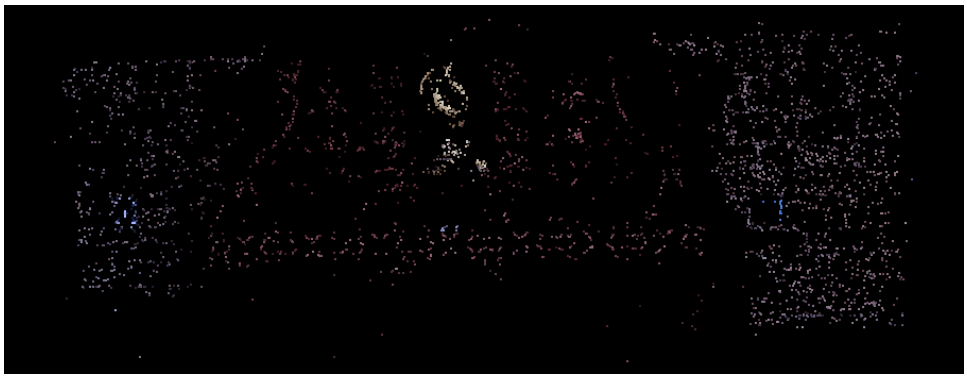

Figure 4.4: 2-View 3D Reconstruction

We will now discuss how the error measure that we set up may help us in finding a more robust reconstruction. First we should note that reprojection means we simply take the triangulated 3D point and reimage it on a camera to get a reprojected 2D point, we then compare the distance between the original 2D point and the reprojected 2D point. If this distance is large this means we may have an error in triangulation, so we may not want to include this point in the final result. Our global measure is the average reprojection distance and may give us a hint to how

our triangulation performed overall. High average reprojection rates may point to a problem with the P matrices, and therefore a possible problem with the calculation of the essential matrix or the matched feature points.

We should briefly go back to our discussion of camera matrices in the previous chapter. Were we mentioned that decomposing the camera matrix P1 from E gives four different solution, but only one composition is correct. Now that we know how to triangulate a point, we can add a check to see which one of the four camera matrices is valid based on the average reprojection error. Next we are going to take a look at recovering more cameras looking at the same scene, and combining the 3D reconstruction results.

4.4.3 Reconstruction from multiple views

Now that we know how to recover the motion and scene geometry from two cameras, it would seem trivial to get the parameters of additional cameras and more scene points simply by applying the same process. This matter is in fact not so simple as we can only get a reconstruction that is up-to-scale, and each pair of pictures gives us a different scale.Next, we add another camera to the process. We select the camera that observes the largest number of tracks whose 3D locations have already been estimated, and initialize the new camera's extrinsic parameters using the direct linear transform (DLT) technique inside a RANSAC procedure as discussed in previous chapters.

Having found an initial structure, we may continue; however, our method requires quite a bit of bookkeeping. First we needs two aligned vectors of 3D and 2D points. Aligned vectors mean that the ith position in one vector aligns with the ith position in the other. To obtain these vectors we need to find those points among the 3D points that we recovered earlier, which align with the 2D points in our new frame. A simple way to do this is to attach, for each 3D point in the cloud, a vector denoting the 2D points it came from. We can then use feature matching to get a matching pair.After finding the Projection matrix for the added camera we then follow simple procedure of triangulation with already inserted camera's .We first find the camera with the greatest number of matches, M, to existing 3D points, then add any camera with at least 0.20M matches to existing 3D points.In the following image we see an incremental reconstruction 3D points :

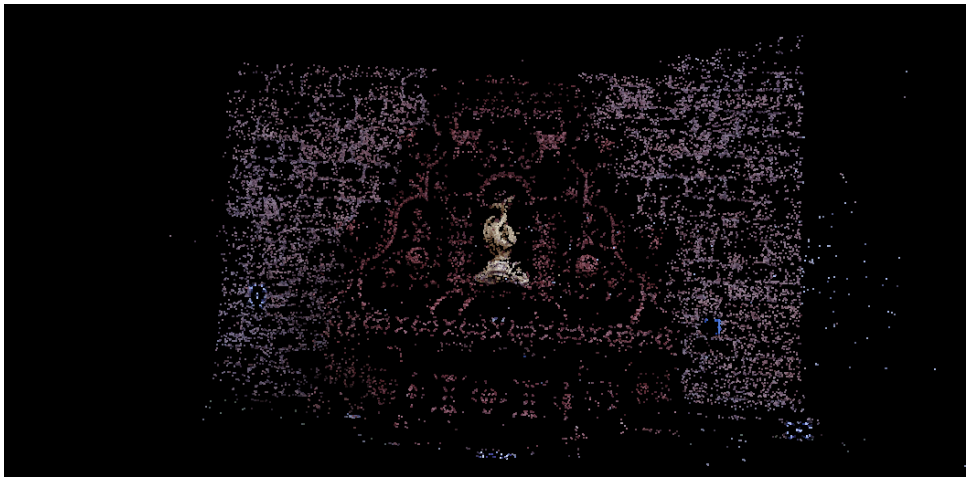

Figure 4.5: 4-View 3D Reconstruction

4.4.4 Visualizing 3D point clouds with Point cloud library

For visualization we haveused an up-and-coming sister project for OpenCV, called the Point Cloud Library (PCL). It comes with many tools for visualizing and also analyzing point clouds, such as finding flat surfaces, matching point clouds, segmenting objects, and eliminating outliers. These tools are highly useful if our goal is not a point cloud but rather some higher-order information such as a 3D model .

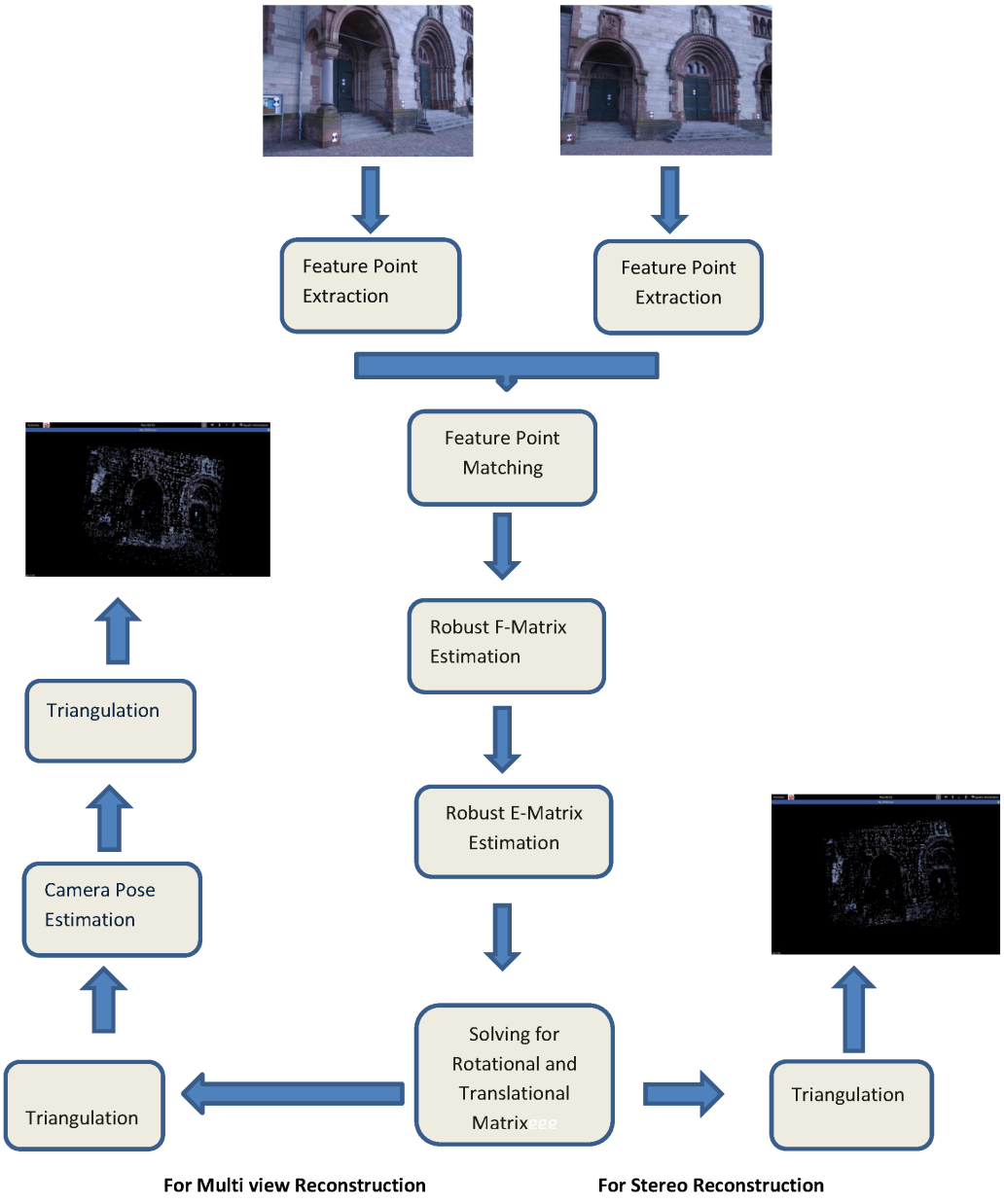

Figure 4.6: Schematic of the adopted strategy

Chapter *5*

Experimental Results

In this chapter we highlight the results of Multiview Stereo that we have obtained for all our three datasets. We have used the PCL (point cloud library) to visualize the 3D points.The performance metric used is the reprojection error, which is equal to the overall average of the reprojection error of all the points in the 3D cloud which is depicted in the following figure.

Table 5.1: Mean Re-Projection Error

No. of Images	Fountain	Herzesu	Castle
2 Images	8.4919	17.5315	14.9132
3 Images	25.6543	31.4902	26.1845
4 Images	51.4562	53.7451	47.3217

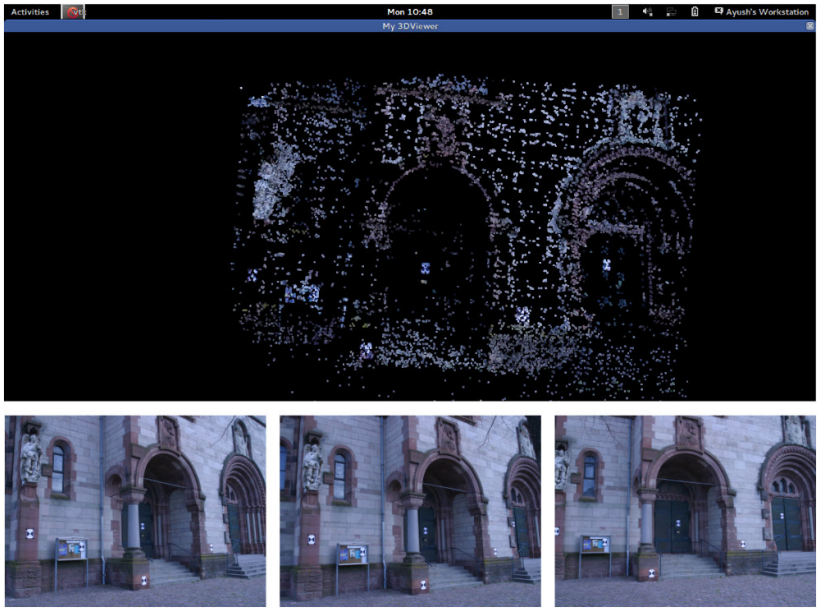

Figure 5.1: 4 Images Reconstruction from Herx-Jesu Dataset

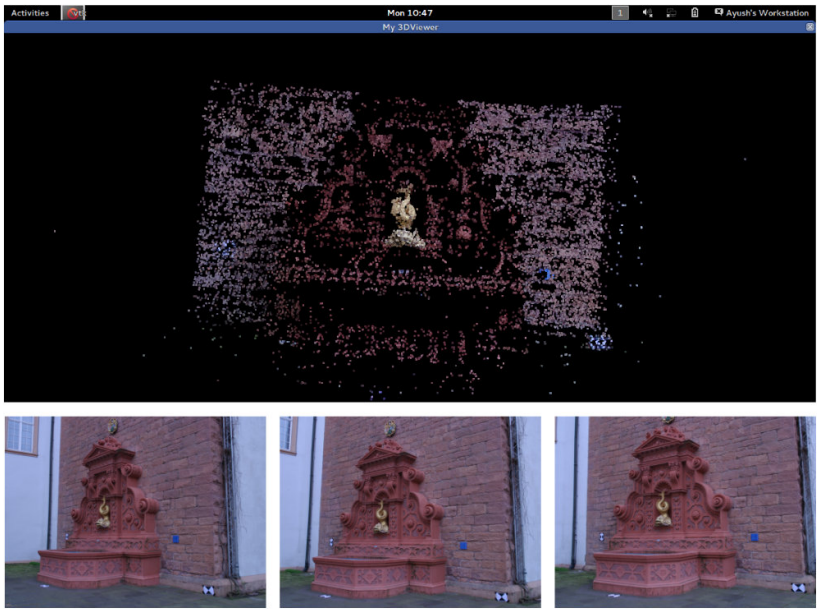

Figure 5.2: 4 Images Reconstruction from Fountain-P12

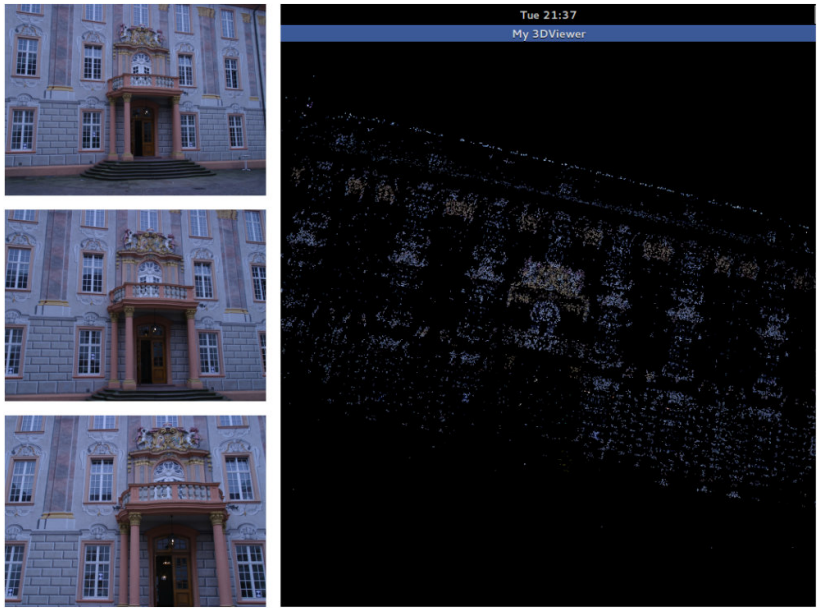

Figure 5.3: 4 Images Reconstruction Castle-Entry P-10

Chapter *6*

Conclusions and Future Work

6.1 Conclusion

However, the state-of-the-art SfM methods are far more complex. There are many issues we choose to disregard in favor of simplicity, and plenty more error examinations that are usually in place. Our chosen methods for the different elements of SfM can also be revisited. For one, Hartley and Zisserman propose a highly accurate triangulation method that minimizes the reprojection error in the image domain. Some methods even use the N-view triangulation once they understand the relationship between the features in multiple images.

6.2 Future Work

We have observed that as we increase the number of input images, the reprojection error also increases. There are many optimisation techniques that can be used to reduce the reprojection problem. Some of these optimisation methods are Bundle Adjustment based methods.

Another work that can be done from here is dense reconstruction. In this work we have implemented sparse reconstruction through SfM but in future we can further get more dense reconstruction.

Bibliography

[1] R. S. D. Scharstein and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Proc. IEEE Workshop Stereo and Multi-Baseline Vision (SMBV 2001)*, pages 891–921, 2001.

[2] C. H. Esteban and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *Comput. Vis. Image Underst.*, 96(3):367–392, Dec. 2004.

[3] C. H. Esteban and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *Comput. Vis. Image Underst.*, 96(3):367–392, Dec. 2004.

[4] P. Fua and Y. Leclerc. Object-centered surface reconstruction: Combining multi-image stereo and shading. *International Journal of Computer Vision*, 16:35–56, 1995.

[5] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.

[6] S. Guan and R. Klette. Dense motion and disparity estimation via loopy belief propagation. In *RobVis'08: Proceedings of the 2nd international conference on Robot vision*, pages 993–1008, 2008.

[7] F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *Proc. Intl. Conf. on Computer Vision*. IEEE, 2007.

[8] M. Isard and J. Maccormick. Dense motion and disparity estimation via loopy belief propagation. IEEE, 2005.

[9] T. Kanade, P. Rander, and P. J. Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE MultiMedia*, 4(1):34–47, Jan. 1997.

[10] R. Klette and S. Morales. "prediction error evaluation of various stereo matching algorithms on long stereo sequences. 2009.

[11] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proceedings of the 7th European Conference on Computer Vision-Part III*, ECCV '02, pages 82–96, London, UK, UK, 2002. Springer-Verlag.

[12] F. Liu and V. Philomin. Disparity estimation in stereo sequences using scene flow. 2008.

[13] D. B. M. Z. Brown and G. D. Hager. Advances in computational stereo. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 993–1008. IEEE, 2003.

[14] M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(4):353–363, Apr. 1993.

[15] S. Roy and I. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem, 1998.

[16] A. G. S. Smirnov. 3d video processing algorithms - part i. 2010.

[17] D. Scharstein and R. Szeliski. The middlebury stereo vision page. [online].

[18] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1:519–528, 2006.

[19] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *International Journal of Computer Vision*, pages 1067–1073, 1997.

[20] J. D. D. S. R. S. Steven M. Seitz, Brian Curless. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR 2006*, 2006.

[21] G. V. L. F. P. T. U. Strecha C., von Hansen W. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *CVPR 2008*. IEEE, 2008.

[22] A. Treuille, A. Hertzmann, and S. M. Seitz. Example-based stereo with general brdfs. In *In European Conference on Computer Vision*, pages 457–469, 2004.