PRIORITIZED 3D SCENE RECONSTRUCTION AND RATE-DISTORTION EFFICIENT REPRESENTATION FOR VIDEO SEQUENCES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EVREN İMRE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2007

Approval of the Thesis:

PRIORITIZED 3D SCENE RECONSTRUCTION AND RATE-DISTORTION EFFICIENT REPRESENTATION FOR VIDEO SEQUENCES

submitted by **EVREN İMRE** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, **Graduate School of Natural and Applied Sciences** _____

Prof. Dr. İsmet Erkmen
Head of Department, **Electrical and Electronics Engineering** _____

Assoc. Prof. Dr. A. Aydın Alatan
Supervisor, **Electrical and Electronics Eng. Department, METU** _____

Examining Committee Members

Prof. Dr. Mete Severcan
Electrical and Electronics Engineering Dept., METU _____

Assoc. Prof. Dr. A. Aydın Alatan
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Levent Onural
Electrical and Electronics Engineering Dept., Bilkent University _____

Prof. Dr. Kemal Leblebicioğlu
Electrical and Electronics Engineering Dept., METU _____

Assoc. Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering Dept., METU _____

Date: **28.08.2007**

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Evren İmre

Signature :

ABSTRACT

PRIORITIZED 3D SCENE RECONSTRUCTION AND RATE-DISTORTION EFFICIENT REPRESENTATION FOR VIDEO SEQUENCES

İmre, Evren
Ph.D., Department of Electrical and Electronics Engineering
Supervisor: Assoc. Prof. Dr. A. Aydın Alatan

August 2007, 201 Pages

In this dissertation, a novel scheme performing 3D reconstruction of a scene from a 2D video sequence is presented. To this aim, first, the trajectories of the salient features in the scene are determined as a sequence of displacements via Kanade-Lukas-Tomasi tracker and Kalman filter. Then, a tentative camera trajectory with respect to a metric reference reconstruction is estimated. All frame pairs are ordered with respect to their amenability to 3D reconstruction by a metric that utilizes the baseline distances and the number of tracked correspondences between the frames. The ordered frame pairs are processed via a sequential structure-from-motion algorithm to estimate the sparse structure and camera matrices. The metric and the associated reconstruction algorithm are shown to outperform their counterparts in the literature via experiments. Finally, a mesh-based, rate-distortion efficient representation is constructed through a novel procedure driven

by the error between a target image, and its prediction from a reference image and the current mesh. At each iteration, the triangular patch, whose projection on the predicted image has the largest error, is identified. Within this projected region and its correspondence on the reference frame, feature matches are extracted. The pair with the least conformance to the planar model is used to determine the vertex to be added to the mesh. The procedure is shown to outperform the dense depth-map representation in all tested cases, and the block motion vector representation, in scenes with large depth range, in rate-distortion sense.

Keywords: Feature tracking, structure-from-motion, rate-distortion efficient scene representation.

ÖZ

VIDEO GÖRÜNTÜLERİ İÇİN ÖNCELİKLENDİRİLMİŞ 3B SAHNE GERİ ÇATIMI VE HIZ-BOZULUM BAĞLAMINDA VERİMLİ GÖSTERİMİ

İmre, Evren
Doktora, Elektrik-Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. A. Aydın Alatan

Ağustos 2007, 201 sayfa

Bu tez çalışmasında, 2B video görüntülerini kullanarak bir 3B sahne geriçatımı gerçekleştiren özgün bir yöntem önerilmiştir. Önerilen yöntemde, öncelikle Kanade-Lukas-Tomasi izleyicisi ve Kalman filtresi yardımıyla sahnedeki belirgin özniteliklerin bir kareler arası yer değişiklikleri dizisi olarak temsil edilen gezingeleri oluşturulur. Ardından, sahneyi görüntüleyen kameralar bir metrik referans geriçatımına göre yerleştirilip, videodaki her kare çiftinin 3B geriçatıma uygunluğunu değerlendirebilmek için temel çizgi ve öznitelik çifti sayısını göz önüne alan özgün bir ölçev hesaplanır. Bu ölçeve göre sıralanan kare çiftleri, sıralı hareketten-yapı temelli bir algoritmayla işlenip, seyrek yapı ve kamera kestirimleri hesaplanır. Önerilen yaklaşımın teknik yazındaki benzerlerinden üstün olduğu deneylerle gösterilmiştir. Son olarak, hız-bozunum açısından verimli, örgü temelli bir 3B sahne gösterimi oluşturulur. Örgü yaratım süreci, bir

hedef görüntü ve onun bir örgü ve bir referans görüntüsünden elde edilen kestirimi arasındaki hata tarafından yönetilir. Hedef ve kesitim arasındaki fark görüntüsü üzerinde yansıması en büyük hatayı içeren üçgensel yama, iyileştirme için seçilir. Bu yamanın hedef ve referans görüntülerindeki yansımaları içinde yer alan öznitelik çiftleri arasından, üçgensel yamaya en az uyumlu olan çift, gösterime eklenecek 3B düğümü hesaplamak için kullanılır. Önerilen yöntem denenen tüm durumlarda sık derinlik haritası metodundan, derinliğin yüksek olduğu sahnelerde ise blok hareket vektörleri metodundan daha iyi sonuçlar vermiştir.

Anahtar kelimeler: Öznitelik takibi, hareketten-yapı, hız-bozulum açısından verimli sahne gösterimi.

ACKNOWLEDGEMENTS

PhD is a long journey that tests the limits of the mental and emotional endurance of those who embark on it. I would like to express my gratitude to the following people for sharing my burden, and any others who were left out in this, by no means complete, list due to a combination of a desire to prevent the acknowledgements grow longer than the dissertation itself, and the fallacy of human memory.

My supervisor, Assoc. Prof. Dr. Aydın Alatan, has been a major driving force behind my studies, with his patience, motivation and vision. He led the way out when I was lost, and made me walk onwards when carrying on seemed impossible.

Egemen İmre, and Miase Bayramoğlu have been with me until the very end of my PhD studies. Almost everything that made the last 5 years bearable can be traced to their unfaltering support through the thick and thin. And the rest, to my parents, for they have been remarkably supportive and patient with me.

Emre Özkan, Umut Orguner, Eren Akdemir, Yücel Özbek, Ülkü Çilek Doyuran, Selva Murat Çürükoğlu, Tülay Akbey, Işıl Yazgan Birinci and Özlem Pasin, my fellow research assistants, made my time in the department enjoyable and memorable through their friendship.

Cevahir Çığla, Osman Serdar Gedik, Sebastian Knorr, Burak Özkalaycı, Uğur Topay, Elif Vural, Mehmet Oğuz Bici, Engin Tola and Assoc. Prof. Uğur Güdükbay cooperated with me in various stages of my PhD studies. I appreciate their diligence in work, and their willingness to share their expertise and resources, to help me to overcome the problems that would otherwise prove insurmountable.

This work is funded by *EC IST 6th Framework 3DTV NoE* and partially funded by TÜBİTAK under Career Project 104E022.

The sequences *Palace, Wall* and *Cliff* are provided by courtesy of LexiTV, and *Stefanie*, by courtesy of Sat1.

TABLE OF CONTENTS

ABSTRACT iv
ÖZ vi
ACKNOWLEDGEMENTS viii
TABLE OF CONTENTS x
LIST OF ABBREVIATIONS xiii
CHAPTERS
1 INTRODUCTON 1
1.1 3D Scene Reconstruction from Uncalibrated 2D Video Sequences 4
1.1.1 Problem Definition 4
1.1.2 Literature Review 5
1.1.3 Overview of the Proposed Solution 11
1.2 Major Contributions 13
1.3 Outline of the Thesis 15
2 FEATURE EXTRACTION AND TRACKING 17
2.1 Feature Extraction 18
2.1.1 Harris Corner Detector 18
2.1.2 Subpixel Refinement 20
2.1.3 Corner Reliability Heuristics 23
2.2 Feature Matching 24
2.2.1 Similarity Metrics 25
2.2.2 Association 28
2.2.3 Guided Matching 31
2.3 Feature Tracking 31
2.3.1 Discrete Kalman Filter 32
2.3.2 Optical Flow Equation and Kanade-Lukas-Tomasi Tracker 33

2.4 Optical Flow-Based Tracking 37
2.4.1 System Model 37
2.4.2 OF Tracker 38
2.5 Corner-to-Corner Tracking 41
2.5.1 System Model 41
2.5.2 CC Tracker 42
2.6 Experimental Results 44
2.6.1 Intensity Tracking 47
2.6.2 Fundamental Matrix Estimation 52
2.7 Conclusion 53
3 SPARSE 3D SCENE RECONSTRUCTION FROM UNCALIBRATED 2D VIDEO SEQUENCES 54
3.1 Image Formation and Reconstruction Ambiguity 55
3.2 Projective 3D Reconstruction from Two-Views 57
3.2.1 Estimation of Fundamental Matrix 58
3.2.2 Triangulation 65
3.3 Metric 3D Reconstruction from Two Views 69
3.4 Self-Calibration 72
3.5 Trajectory Segmentation 77
3.6 Multi-View 3D Reconstruction 79
3.6.1 Sequential Reconstruction 79
3.6.2 Bundle Adjustment 82
3.7 Prioritization 87
3.8 Prioritized Sequential 3D Reconstruction 90
3.9 Experimental Results 96
3.9.1 Different Prioritization Metrics 97
3.9.2 Multiple Initial Frames 102
3.9.3 Prioritized vs. Conventional Sequential Reconstruction 103
3.10 Conclusion 106
4 EFFICIENT PIECEWISE PLANAR SCENE REPRESENTATION 108
4.1 Design Considerations 109
4.2 Surface Interpolation with Delaunay Triangulation 112

4.3 Rendering a View of the Scene 116
4.4 Rate-Distortion Efficient Piecewise Planar Scene Reconstruction 120
4.4.1 Sequential Phase 120
4.4.2 Nonlinear Optimization 124
4.5 Experimental Results 129
4.5.1 Piecewise Planar Reconstruction Experiments 129
4.5.2 Stability of the Convergence Point 162
4.5.3 Coarse-to-Fine vs. Fine-to-Coarse 162
4.5.4 Vertex Selection: Geometry vs. Image Error 166
4.5.5 Rate-Distortion Performance 168
4.6 Conclusion 177
5 SUMMARY AND CONCLUSION 178
5.1 Summary and Conclusions 178
5.2 Future Directions 182
REFERENCES 185
VITA 199

LIST OF ABBREVIATIONS

BMV: Block motion vector

CC: Corner-to-corner (tracker)

E-Matrix: Essential matrix

F-Matrix: Fundamental matrix

GRIC: Geometric robust information criterion

KLT: Kanade-Lukas-Tomasi (tracker)

MFSfM: Multi-frame structure-from-motion

NC: Neighborhood constraint

NCC: Normalized cross-correlation

OF: Optical flow (tracker)

RANSAC: Random sample consensus

SfM: Structure-from-motion

SSD: Sum of square differences

STE: Symmetric transfer error

TSE: Total square error

CHAPTER 1

INTRODUCTION

The relation between a 3D real world scene and its images has been studied for centuries by painters, photographers, psychologists, mathematicians, physicists and engineers. The interest from such an impressive array of disciplines stems from the broad variety of applications, ranging from cartography to data visualization and entertainment. However, among these, entertainment applications have been one of the major driving factors behind the research in this area, with a history dating back to the invention of stereoscopic photography in 1838. Moreover, the improvements in 3D display technologies, and the increasing affordability of auto-stereoscopic displays for individual desktop use will certainly accentuate the already influential role of entertainment applications, due to increased 3D content demand.

The third dimension, despite its exciting potential, cannot yet challenge the dominance of the 2D media in visual entertainment, due to established distribution channels and widespread use of 2D display devices, such as television, monitor and movie screen. This fact naturally limits the production and choice of available of 3D content. In the future, this problem will be overcome by the positive feedback loop between the increasing demand created by 3D displays and 3D content for each other. Today, this limitation can be overcome by tapping into the

immensely rich repository of 2D media, by devising methods to convert it to 3D. Such methods have two immediate applications:

- **3D TV:** Online conversion of 2D TV broadcast to 3D enables a 3D TV implementation that is totally compatible with the existing 2D content production facilities and broadcast network.
- **Recycling existing material:** The first motion picture was shot in 1888, and in 2005, only in USA, 611 movies were produced. An off-line conversion technology for converting the existing 2D movies and archive material into 3D can inject tens of thousands of titles into an otherwise content-starved field.

It is these potential applications, as well as the promise of an interesting research, that prompted the author to choose 3D conversion of 2D video as the topic of his PhD studies.

The exact problem definition and a framework for solution are reserved for the following sections. Basically, the fundamentals of the solution remains the same as the template laid out by the other major works in this area, such as [1], [2] and [3], whose essential components are feature extraction and matching, sequential sparse reconstruction, self calibration and dense reconstruction blocks. The difference between this work and the literature lies in the shift of emphasis to different aspects of the problem, such as:

- **Feature extraction and matching for video:** With regards to feature extraction and matching, video sequences have two important properties that should be exploited: Lighting and feature locations in consecutive frames do not change substantially, i.e., there exist translation-invariant features in the scene. However, in [2], since frame-collections are also designated as possible

inputs, affine-invariant features are used. This flexibility comes at a price, as in video case, enforcement of unnecessary invariance conditions reduce the number of matches [4]; hence, the quality of the camera matrix and sparse structure estimates. In [1], corner features in consecutive frames are paired together to form long trajectories. In Chapter 3, this approach is shown to be inferior in the number and suitability of the recovered features for 3D reconstruction, to a Kanade-Lukas-Tomasi tracker [16], a method employed in [3] and this work.

- **Frame pair prioritization:** Video sequences do not suggest an inherent processing order that can both achieve a good reconstruction, and avoid degenerate and numerically unstable cases. Therefore, it is important to be able to automatically select the frame pairs which offer accurate and informative estimates of the scene and to establish a processing order, to ensure the convergence of the sequential reconstruction algorithm to a good solution. However, this problem is not mentioned at all in [1] and [3]. In [5], a work closely related to [2], the problem is recognized and solved by using geometric robust information criterion (GRIC) a metric defined in [6]. In this work, another solution is proposed, and shown to be superior to GRIC.

- **Efficient scene representation:** For dense scene representation, dense depth maps are employed in [1] and [2]. In [3], as in this work, a triangular mesh is chosen to better utilize the available sparse structure estimate. However, in all works sharing the scope of this thesis, only accuracy is emphasized. In this dissertation, dense scene representation is studied in the framework suggested in [92], i.e., as a rate-distortion problem, to obtain a not only accurate, but also efficient representation of the scene, to facilitate its transmission and storage.

In the following sections, the problem, a review of the relevant literature and the basic solution approach is presented.

1.1 3D Scene Reconstruction from Uncalibrated 2D Video Sequences

1.1.1 Problem Definition

The "3D reconstruction from uncalibrated 2D video" problem can be formally defined as estimating a dense 3D representation of the scene from an uncalibrated video sequence, under the assumptions of rigid body motion, and the existence of camera motion. The emphasis on *uncalibrated video* is to signify that no prior information about the data is available, a situation commonly encountered when processing video sequences acquired from TV broadcast, or 2D archive material. The problem is a special case of *multi-view 3D reconstruction*, with the following distinguishing features:

- **Video sequence is the only source of information:** No information on the cameras or the scene is available, except for what can be extracted from the video sequence (e.g., no calibration information, auxiliary sensor information for camera motion, actual scale of the scene, parallelism or perpendicularity).
- **No control on data acquisition:** Only passive sensors are used and neither camera parameters nor camera motion can be controlled. Moreover, there is a wide variety of scenes to be dealt with, ranging from textured natural outdoor scenes and urban scenes dominated by planes and regular texture, to indoor scenes with mostly flat, non-textured walls.
- **Dynamic scene:** In addition to a still background, a scene may contain dynamic elements, i.e., independently moving objects. While it is possible to isolate these elements and reconstruct them individually, scale ambiguity prevents a precise localization in the coordinate system of the background automatically. Still, it is possible to limit the uncertainty to some extent by using occlusions and disocclusions.

- **Causality:** In an on-line 3DTV application, the data should be processed causally or at most with a small processing delay. However, conversion of archive material is amenable to non-causal processing.

- **Computational load:** 2D-3D conversion for 3DTV requires real-time operation. However, conversion of archive material allows off-line computation. This fact, coupled with the availability of non-causal processing techniques, means that the conversion of archive material is a relatively less challenging problem when compared to 3DTV.

- **Supervision:** The computer vision field offers the necessary tools to completely automate the 2D-to-3D conversion chain for static scenes [10]. However, all these tools have their domains of validity, defined by their fundamental assumptions on the input. A simple way to overcome this limitation is to employ a man-in-the-loop, who makes the critical decisions, and delegates the computationally intensive procedures to the computer. These decisions include, but not limited to, identifying frames that give rise to certain special cases, identifying parallelisms and perpendicularities in the scene and placing dynamic elements [10].

The system proposed in this work primarily focuses on the former two of these issues. A solution for the dynamic scene case is included for sake of completeness, but the proposed method is basically for static scenes. The design choices were often influenced by their implications on the computational load however, real-time performance was never pursued. Finally causality was not considered as a design constraint, and the possibility of supervision is forsaken in favor of a fully-automatic system.

1.1.2 Literature Review

As mentioned above, solution approaches to 2D-3D conversion problem are typically composed of feature tracking, self-calibration, sparse and dense

reconstruction modules. Moreover, when dynamic scene is a possibility, the system should be enhanced with trajectory segmentation capability, to associate the trajectories with the correct static or dynamic scene elements. Each module performs a specific task, and is the subject of a distinct field of research, hence, deserves an individual review of the relevant literature.

Feature Matching and Tracking

Typical scene features used in 3D reconstruction are corners, for a number of reasons including the existence of mature and robust algorithms for their extraction, matching and tracking, and unambiguous localization on the image. Besides, line features, the closest rival of corners, have more severe degenerate cases in reconstruction [10].

The basic approaches for feature extraction are using curvatures [13] and finding the maxima of nonlinear transformations of the image gradient. The latter group includes very successful and popular feature detectors, such as Harris corner detector [7] and SIFT [8].

The choice of matching/tracking algorithm depends on the relative calibration between images, such as orientation and scale variations, and the available computational resources. A general rule of thumb can be stated as follows: The more complex a matching procedure is, the smaller, but more reliable the established correspondences become. In order to determine matching features, either intensity similarity (e.g., cross correlation or affine warping) [15], or structural similarity (e.g., neighborhood constraint) [14] is utilized. In video case, KLT tracker successfully constructs feature trajectories [9][16]. A typical subsequent step is the elimination of possible outliers via epipolar criterion [10].

An extensive survey of different approaches to corner detection and matching problem can be found in [4].

Trajectory Segmentation

The solution approaches for the trajectory segmentation problem can be classified into four categories. Optical flow-based methods assume a scene that is composed of planes at various depths, and utilize a simple clustering to achieve the desired segmentation [17]. Another set of solutions utilizes eigen decomposition of the *affinity matrix*, a structure which contains the similarity information among all features [19]. Geometric methods exploit the constraints imposed by the epipolar geometry and the rigid body motion assumption. While the epipolar constraint and the fundamental matrix (F-matrix) is a popular choice [20][21], more general model selection-based methods are also available [6]. Finally, statistical techniques such as *sequential importance sampling* also have a niche in this field [18]. Among these methods, the geometric approach enjoys a popularity stemming from its simplicity and compatibility with the 3D nature of the problem.

Self-Calibration

The first and perhaps the best-known self-calibration technique is developed by Fagueras et al.[22], and involves the solution of *Kruppa equations* [23]. In [24], an indirect approach that upgrades a projective model first to affine, and then to Euclidean stratum is employed, by utilizing the *modulus constraints*. The constraints on the singular values of essential matrix (E-matrix) give rise to a relatively robust technique, presented in [27]. Recently, a method that utilizes the *cheiral inequalities* [10] to achieve a quasi-affine reconstruction, which is then upgraded to metric stratum, is also proposed [28].

Another class of solutions constrains or fixes the unknown parameters, or motion. One such technique, a remarkably simple and stable one, is described in [2], in

which, soft constraints are used to weight a linear equation system. In [25] and [26], recovery of the unknown focal length by using only two views is shown to be possible, through a solution of a combination of linear and non-linear equations. A recent technique employs *Größner bases* to construct a 15^{th} degree polynomial, solution of which yields the unknown focal length from two cameras [29].

Sparse Reconstruction

Two-view structure-from-motion (SfM) problem has been studied for 25 years, beginning with the seminal work of Longuet-Higgins [31], and a complete solution for metric case was proposed as early as 1989 by Weng et al [30]. However, many aspects of the problem were further studied to yield improvements in F-matrix estimation and triangulation. The highlights of the two-view SfM research in the following years are the 7-point [34], and normalized 8-point algorithms [12], robust F-matrix estimation via stochastic optimization techniques, specifically RANSAC [33], characterization of the uncertainty of F-matrix [35] and polynomial triangulation. [36]. The recent research in the field is more focused on better stochastic optimization schemes that are capable of utilizing the correspondence reliability [38][86], or that can deal with the degenerate cases [39], and efficient estimation procedures [37]. An excellent review of F-matrix estimation techniques can be found in [34] and [10] includes an extensive treatment of the subject, complete with theoretical and practical aspects.

On the other hand, the solution approaches to multi-frame extension of the SfM problem (MFSfM) can be categorized into batch and sequential algorithms. Batch algorithms attempt to solve the MFSfM problem by processing all available data at once. Their best known example is the *factorization method*, in which, the rank constraint on *trajectory matrix* is exploited to factorize it into two terms, corresponding to camera orientation and structure [40]. The algorithm is first

designed for orthographic projection, and later improved to deal with other camera models [41][56], articulated motion [43], and independently moving objects [42]. The most notable shortcoming of factorization algorithms is their inability to utilize partial trajectories. Therefore, to obtain a relatively populous sparse point cloud, it is necessary to complete the partial trajectories [44]. Finally, the error analysis of this algorithm is presented in [45]. Another well-known approach for MFSfM problem is *bundle adjustment*, a technique that attempts to find the optimal structure and camera matrix estimates by performing a *Levenberg-Marquardt* minimization of the reprojection error over these parameters [49]. While efficient algorithms exist to reduce the computational cost [46][47], and to improve its stability [48], this method is known to be extremely sensitive to initial estimate [49].

While sequential algorithms initially comprised a separate branch, currently, it is common practice to use them to find a good initial estimate for bundle adjustment (or equivalently, to refine their results with bundle adjustment). The first sequential algorithms were formulated via extended Kalman filter [51][52], to estimate the unknown state vector composed of camera and structure parameters, by using the 2D correspondences, or essential matrices as observations [55]. However, these algorithms require accurate initial estimates. A similar method employs *particle filters* to estimate the posterior probability density of the state vector, given the observations [53]. The estimate-fusion approach leads to sequential algorithms that integrate two- or multiple-view sub-estimates of camera and structure parameters [2][57]. With a proper weighting scheme, this class of algorithms yields successful results. The bias and variance of sub-estimates [53] can be used to determine such a weighting scheme.

Dense Reconstruction

A dense 3D reconstruction can be described either by a point-based representation, as a depth-map defined on the same lattice with the reference

frame, or as a mesh-based piecewise planar surface, or by a volumetric representation, such as voxels [58]. A good review of voxel based and depth-map based methods can be found in [58] and [59]. Depth map-based representations have the advantage of exploiting the existing image and video coding techniques for compression. However, multiple depth-maps of the same scene from different views have an inherent redundancy due to overlapping parts, making other 3D representations more desirable. One of these 3D representations, voxels, is known to achieve high resolution, whereas demanding in terms of the computational resources. On the other hand, piecewise planar representations offer an efficient alternative for many man-made and natural real world scenes that can be well-approximated by planes. Besides, a piecewise planar representation is a natural extension of a sparse point cloud, therefore, facilitate interaction between the other modules of the reconstruction chain.

The considerable body of research on piecewise planar scene representations can be presented in two major classes. In the first approach, a planar surface is fit onto an irregular 3D point cloud. A good example is presented in [61], in which the point cloud is divided into cells and a dominant plane is identified in each cell via RANSAC. An equivalent procedure is described in [60] to determine the homographies induced by scene planes from 2D correspondences.

The use of triangular meshes, specifically *Delaunay triangulation*, due to its certain optimality properties [62] and compact representation as a sequence of vertices [77], characterizes the second approach. There exist successful algorithms that can construct a triangular mesh from an irregular 3D point cloud [63]. However, image-based triangulation (IBT) techniques [64] are one step beyond, as they are also capable of incorporating the intensity information. The basic algorithm utilizes edge swaps on a triangular mesh, to minimize the intensity prediction error of an image of the scene, acquired by a known camera matrix [64]. In [65], a simulated annealing procedure, that is equipped with a rich arsenal

of tools in addition to edge swap, is employed. In the algorithm proposed in [66], a similar idea is used to represent a disparity map. However, it differs from the others by adding vertices to locations where the prediction error is largest, instead of simplifying a complex mesh.

1.1.3 Overview of the Proposed Solution

The input to the proposed system is an uncalibrated 2D video sequence depicting, preferably, a static scene and the output is the dense 3D reconstruction of the scene observed in the sequence, represented as a mesh. The first step of the reconstruction chain is the establishment of feature correspondences between the frames. To this aim, in each frame, salient features are extracted by Harris corner detector [7], and tracked by KLT tracker [9][16] that is assisted by a Kalman filter. This module also determines the key-frames -frames of a video sequence with significant 3D information content- using GRIC. These frames are later used in segmentation, self-calibration and sparse reconstruction modules.

The next step is the segmentation of the trajectories into sets corresponding to the static and dynamic elements in the scene. This can be performed via geometric means, by utilizing the fact that for each rigid 3D motion in the scene, there is a corresponding *fundamental matrix* [10], and each fundamental matrix (F-matrix) defines an *epipolar constraint* [10] for the corresponding motion. If a feature pair belongs to a rigid motion, it should conform to the associated epipolar constraint. Therefore, it is possible to label the trajectories by successively estimating a sequence of fundamental matrices from the feature pairs rejected by the previous iteration. Fundamental matrix estimation is performed by *normalized 8-point algorithm* [12] and *RANSAC* [11], as discussed in [10]. This block is included for sake completeness, and demonstrated to work in synthetic and controlled sequences [85].

Once the motion segmentation is complete, the next stage is the estimation of the internal calibration parameters via the key-frames that are determined in the segmentation module. This is accomplished using the linear self-calibration algorithm described in [2].

The self-calibration stage is followed by sparse reconstruction, individually for the static and the dynamic elements of the scene (if any exists). To this aim, tentative pose estimates are computed for all frames, and camera locations and the number of corresponding feature pairs are used to determine a processing order for all available pairs. Then, a projective sequential reconstruction algorithm, along the lines of the one described in [2], is employed. However, the proposed algorithm is more sophisticated than its progenitor, as it is able to maintain multiple reconstructions instead of a single one. Each of these reconstructions are automatically created, propagated, and merged with each other. If a metric reconstruction is desired, the projective camera matrix estimates can be used to further refine the internal calibration parameters of the camera.

The final stage of the proposed system is the dense reconstruction of the scene. This is performed by gradually building a mesh-based piecewise-planar representation by using the sparse reconstruction and camera matrix estimates that are supplied by the previous stage. This module is capable of operating in the projective stratum, and seeks to obtain a rate-distortion efficient representation. The algorithm is designed to explore the promise of obtaining a rate-distortion efficient representation for a piecewise planar reconstruction; hence, it is only capable of processing static scenes. The representation is obtained by minimizing the intensity error between a frame and its prediction from another frame, therefore the algorithm has also applications in stereo image coding.

The proposed system requires user interaction only to place the (sparse) reconstructions of the dynamic elements in the scene; therefore, it is fully

automatic for the static scenes. Figure 1.1 is a graphical illustration of the proposed system.

Before concluding the overview, it should be emphasized that while the proposed system accommodates for the dynamic scenes, its focus is definitely on static scenes. The sparse reconstruction stage lacks the means to automatically place the dynamic elements in the scene. Moreover, the dense reconstruction stage does not have the capability to segment out the image parts corresponding to the dynamic elements of the scene, hence to produce a mesh-based representation for them. Therefore, it should be kept in mind that the algorithm has only a not-completely-materialized promise of handling dynamic scenes. The work included on this topic is not beyond a preliminary study.

1.2 Major Contributions

Major contributions of this thesis to the existing body of knowledge can be summarized as follows:

- **Prioritized Sequential Reconstruction [82]:** A novel sequential sub-estimate fusion algorithm is proposed for sparse reconstruction. The algorithm is capable of assessing all frame pairs in a video sequence according to their information content and amenability to 3D reconstruction, to establish a processing order. The algorithm maintains multiple sequential reconstructions, and supervises the progress and fusion of each reconstruction.
- **Rate-Distortion Efficient Piecewise Planar Scene Representation [83]:** A rate-distortion efficient piecewise planar dense scene reconstruction algorithm for static scenes is proposed. The algorithm features a coarse-to-fine approach to generate a mesh, by starting from an 8-point mesh and

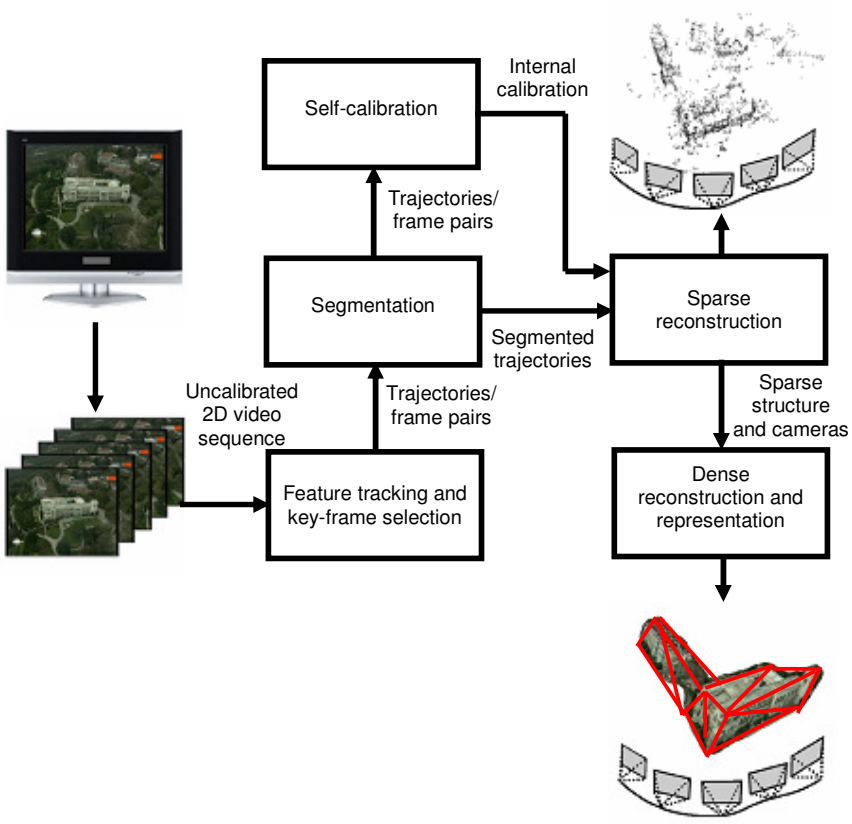


Figure 1.1: Block diagram of the proposed system

refining it at regions where the distortion is highest. The distortion is measured by the error between an image of the scene and its prediction. The coupling of mesh refinement with distortion enables rate-distortion efficient operation. The algorithm operates in projective stratum for added robustness.

1.3 Outline of the Thesis

The organization of the thesis follows the overview in Section 1.1.3. In Chapter 2, feature extraction, matching and tracking processes are explained, and the assumptions and models are presented. In order to solve the track-feature association problem, *auction* [67], a well-known tool in radar tracking literature is introduced. Finally, various performance evaluation metrics are discussed and different tracking approaches are experimentally compared. The material presented in this chapter is not specific to MFSfM problem.

Chapter 3 presents the work towards the solution of MFSfM problem. To this aim, first, two-view and multi-view 3D reconstruction methods are discussed. Then, trajectory segmentation and self-calibration problems and their solutions are briefly mentioned. The *frame pair prioritization* problem is presented, and a solution is proposed. Finally, *prioritized sequential 3D reconstruction,* a novel sequential sub-estimate fusion sparse 3D reconstruction algorithm is described. The chapter is concluded with experiments on various prioritization alternatives.

In Chapter 4, rate-distortion efficient scene representation problem is introduced and an algorithm capable of constructing such a representation for static scenes via Delaunay triangulation is proposed. Various design decisions are experimentally validated, and the representation is compared with its alternatives, dense depth map and block motion vectors, in terms of rate-distortion performance.

Chapter 5 concludes the dissertation with a summary of the work done, a discussion of the results and pointers for future research.

CHAPTER 2

FEATURE EXTRACTION AND TRACKING

SfM techniques for 3D reconstruction problem invariably require identification of 3D scene points (landmarks) in the images of the scene, to estimate the relative orientations of the cameras observing these landmarks, and the positions of the landmarks themselves. These 3D scene points should be distinguishable enough for accurate localization of their projections on the image plane, i.e., the corresponding image features. Besides, they should be easily identifiable from considerably different viewpoints, as, loosely, dissimilar views allow more accurate camera pose and structure estimates. In video case, the latter requirement is less stringent, as it is often possible to track a feature in a sequence of slowly changing views. However, this task still requires landmarks that generate features distinct enough for unambiguous matches across these views. Therefore, it is essential to determine a set of salient features that can be precisely located and accurately tracked throughout the video sequence.

The merits of corners as 2D image features were mentioned in Chapter 1. *Harris corner detector* is a mature algorithm for this task, hence is employed in this work. As for tracking, it can be accomplished in two ways, either by *corner-to-corner* tracking, or any optical flow estimation method, such as Kanade-Lukas-Tomasi tracker (KLT). The former involves associating the corners in each frame with those in the next frame, and chaining these associations together into

trajectories. The latter employs optical flow equation to estimate the displacements of the corners in successive frames. The following section first introduces the elementary building blocks for the feature tracking problem. Then, an example of each tracking approach is presented and their relative performances are experimentally analyzed.

2.1 Feature Extraction

2.1.1 Harris Corner Detector

Harris corner detector [7] models a corner as a point with low self-similarity, i.e., there should be a strong dissimilarity between an image patch centered on a corner and on its neighbors. A common dissimilarity measure is *sum-of-squared-differences* (SSD), defined for discrete images as

$$SSD = \sum_{i,j \in N} \left(I(i-x, j-y) - I(i-u, j-v) \right)^2, \tag{2.1}$$

where N denotes the support of the patch, I, the image, (x,y) and (u,v), the centers of the patches. The *Hessian* of SSD equals

$$\mathbf{C} = \begin{bmatrix} \sum_{i,j \in N} w(i,j) I_H(i,j)^2 & \sum_{i,j \in N} w(i,j) I_H(i,j) I_V(i,j) \\ \sum_{i,j \in N} w(i,j) I_H(i,j) I_V(i,j) & \sum_{i,j \in N} w(i,j) I_V(i,j)^2 \end{bmatrix}, \tag{2.2}$$

where I_H and I_V are the image gradients in the horizontal and vertical directions, and w is a weighting kernel, typically Gaussian for isotropic operation. The image gradients are often computed by the *Sobel operator*, a high-pass filter defined as

$$\mathbf{S_H} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \mathbf{S_V} = \mathbf{S_H^T} \tag{2.3}$$

for horizontal and vertical directions, respectively.

C matrix in Equation 2.2 is known as *cornerness matrix* and its eigenvalues, λ_1 and λ_2, have the following properties:

- If both eigenvalues are small, the point in question is not a significant image feature.
- If only one of the eigenvalues is large, the point is on an edge.
- If both eigenvalues are large, the point is a corner.

A *cornerness* metric that reflects the above observations is defined as

$$\begin{aligned} cornerness &= \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 \\ &= |\mathbf{C}| - kTrace(\mathbf{C})^2. \end{aligned} \tag{2.4}$$

In the original work [7], the value 0.04 is suggested for k for detecting corner features.

In order to locate the corners, Harris corner detector evaluates the cornerness metric at each pixel. If a pixel has a cornerness score above a certain threshold, and it is the local maximum in the cornerness domain within its neighborhood, it is declared as a corner. The size of the neighborhood should not be chosen smaller than the weighting window w, to prevent strong corners from pulling weak maxima above the cornerness threshold, giving rise to false detections. A corner detection example is illustrated in Figure 2.1.

2.1.2 Subpixel Refinement

The original algorithm evaluates cornerness scores at pixels on an integer grid; therefore, it has only pixel-level (integer) resolution. However, SfM problem demands a finer resolution. Subpixel resolution can be achieved via interpolating for the intermediate values in the cornerness domain, to find the local maximum more precisely. Bi-quadratic polynomials offer an easy and robust way to implement this interpolation in a patch around the integer-resolution maximum.

A bi-quadric polynomial is an expression of the form

$$I_C(x, y) = ax^2 + by^2 + cxy + dx + ey + f, \tag{2.5}$$

where I_C stands for the image keeping cornerness scores. The maximum of this surface is located at

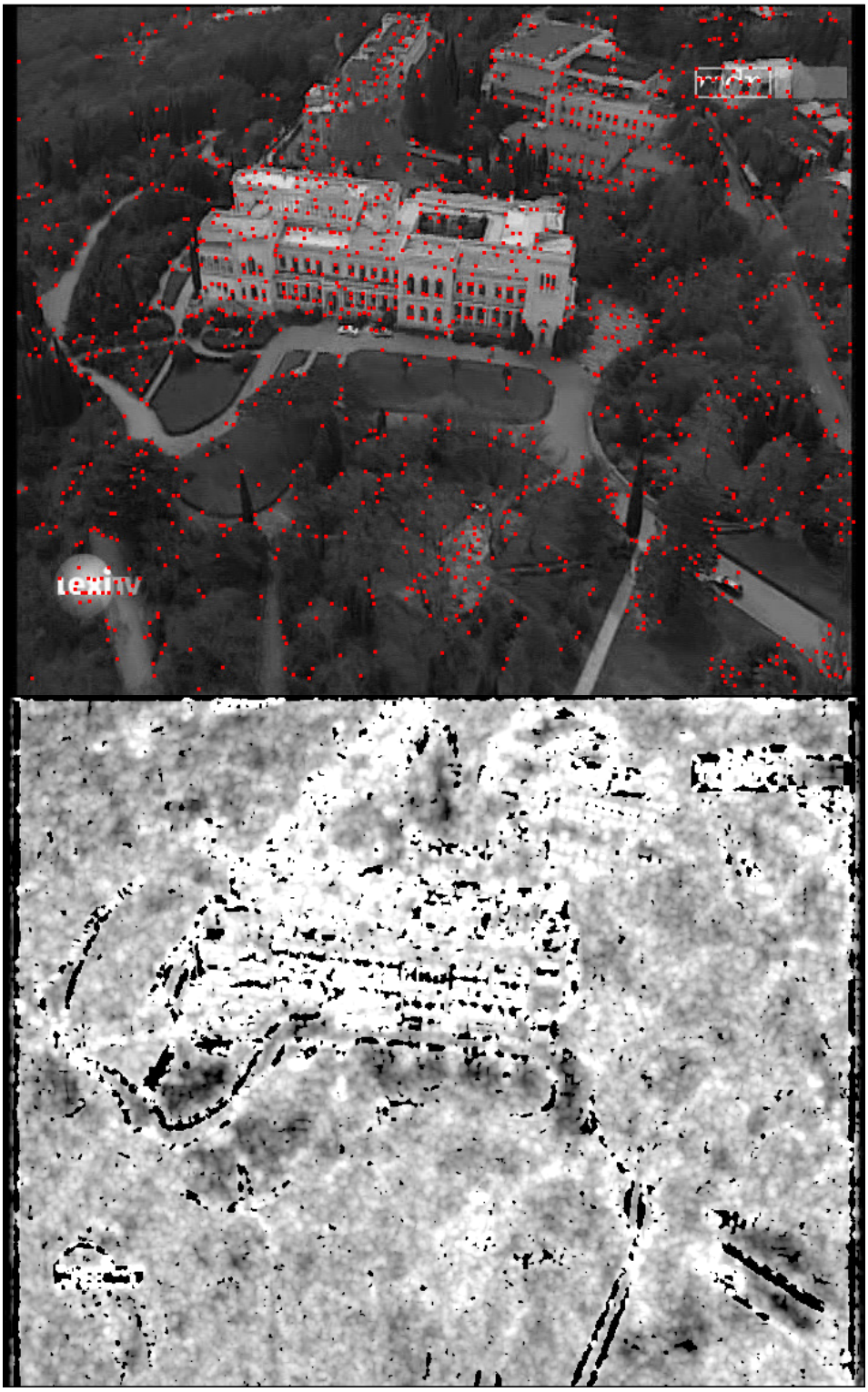


Figure 2.1: Corner detection. *Top*: Corners. *Middle*: Cornerness image. Cornerness values are logarithmically scaled.

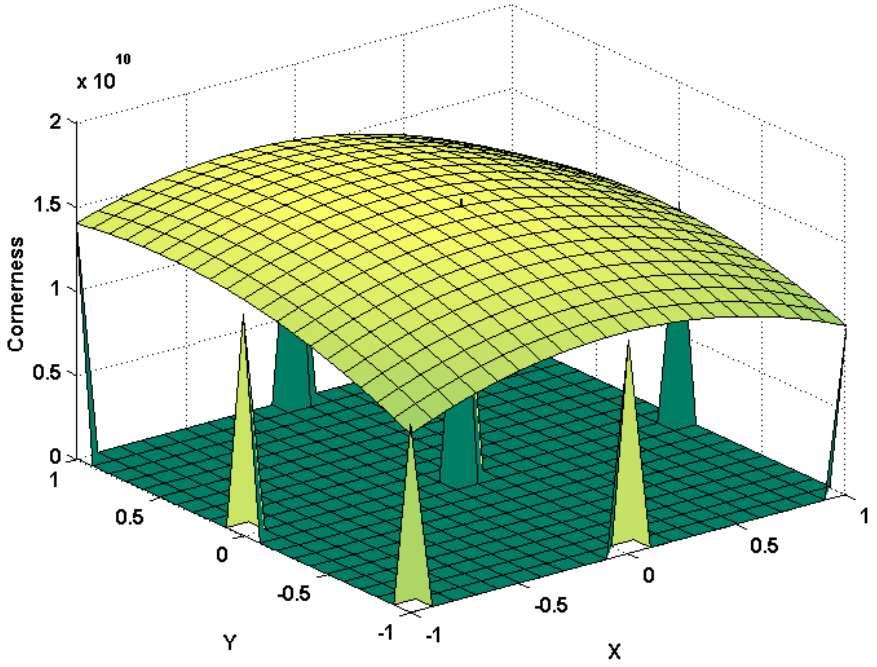


Figure 2.2: Interpolation of the cornerness values. The spikes correspond to the cornerness values on the integer grid, and the surface is the bi-quadric polynomial fitted to these values.

$$x_M = \frac{ce - 2bd}{4ab - c^2}$$
$$y_M = \frac{d - 2ax_M}{c}. \tag{2.6}$$

Even six cornerness values on the integer-resolution grid are sufficient to compute a unique model, whereas more equations improve the accuracy of the model parameters. On the other hand, a bi-quadric polynomial is not guaranteed to be a satisfactory model of local cornerness values, and in larger patches, the probability of model failure increases. In a successful interpolation, the sub-pixel-resolution maximum should lie reasonably close to the pixel-resolution maximum. Figure 2.2 depicts a sample bi-quadric fit.

2.1.3 Corner Reliability Heuristics

Cornerness score is a straightforward way to evaluate the quality of a corner. However, the corner extraction procedure offers some more heuristics to measure reliability, supplementary to naïve thresholding. During the course of the work, the following heuristics emerged as useful.

- **Cornerness contrast:** Cornerness contrast is defined as the difference between a local maximum in cornerness domain and the second largest cornerness value in its neighborhood. A high contrast indicates the insensitivity of the location of the maximum to the changes in the intensity values. Therefore, this heuristic corresponds to noise margin in the pixel-resolution corner localization.
- **NMSE:** This measure is defined as the normalized mean square error (NMSE) of the bi-quadric polynomial model for local cornerness, evaluated at integer grid and normalized by the total energy of the surface

patch fitted to the grid. It measures the accuracy of the subpixel-resolution corner localization.

- **Condition number of the cornerness matrix:** Computation of the optical flow involves the solution of a linear equation, whose coefficient matrix is identical to **C**, given the same neighborhood radius. Therefore, the fitness of the corner for tracking purposes can be inferred from the condition number of this matrix. Condition number indicates the stability of the solution under perturbations, possibly caused by noise; hence, the reliability of the optical flow estimates.

Below is a summary of the Harris corner detection algorithm, as employed in this work.

`Algorithm: Harris Corner Detector`

Input: Image on which the corner extraction is to be performed.

Output: Corner locations.

1. Compute the vertical and the horizontal gradients.
2. Evaluate the cornerness score at each pixel.
3. Determine the maxima above cornerness threshold.
4. Refine the maxima to subpixel-resolution
5. Eliminate the maxima failing the reliability heuristics described in Section 2.1.3

2.2 Feature Matching

Given two sets of features belonging to two frames, a feature matching algorithm seeks to associate the elements of one set with those of the other, while maximizing a quality measure for the entire assignment. The assignment is subject

to the constraints that only one-to-one assignments are allowed and only a subset of all possible associations is admissible. A solution to this problem is characterized by the quality metric, and the association strategy. Both of these issues are discussed in the following sections.

2.2.1 Similarity Metrics

Normalized Cross-Correlation

Normalized cross-correlation (NCC) between two image patches is defined as [68]

$$\rho_{ij} = \frac{E\{(I_i - \mu_i)(I_j - \mu_j)\}}{\sigma_i \sigma_j}, \tag{2.7}$$

where I_i and I_j denote the image patches, μ, their means, and σ, their standard deviations. NCC simply measures the intensity similarity between the two image patches. Assuming that the intensity values of a feature and its neighborhood are essentially constant, a high NCC score often implies a good match. NCC performs well in case of translational motion, but it is not robust to repetitive texture, rotation and affine deformations.

If the images are taken from similar positions, such as successive frames of a video sequence, the positions of the corresponding features are likely to be close. NCC can be enhanced with a proximity term to accommodate for this observation in the similarity assessment. The proximity between feature *i* and feature *j* can be defined as

$$\kappa_{ij} = \frac{1}{1+d_{ij}}, \tag{2.8}$$

where d_{ij} stands for the Euclidean distance between two points from different sets.

The final similarity metric is

$$S_{NCC}(i,j) = \rho_{ij} + \kappa_{ij}. \tag{2.9}$$

Neighborhood Constraint

Consider the features *i* and *j* in the first and second sets, respectively. Let *N(i)* be the set of features within a certain neighborhood of feature *i*. *N(j)*is defined similarly for feature *j*. Let feature *k* be a member of the first set, and feature *l* be a member of the second set. Finally, let ρ_{ij} and ρ_{kl} denote the NCC between the relevant features. The similarity between feature *i* and *j* can be defined as [34]

$$S_{NC}(i,j) = \rho_{ij} \sum_{k \in N(i)} \max_{l \in N(j)} \frac{\rho_{kl}\delta(i,k;j,l)}{1+dist(i,k;j,l)}, \tag{2.10}$$

where

$$\begin{aligned} dist(i,k;j,l) &= \frac{\left|d_{ik}+d_{jl}\right|}{2} \\ r &= \frac{\left|d_{ik}-d_{jl}\right|}{dist(i,k;j,l)}. \end{aligned} \tag{2.11}$$

In this equation, *d* represents the Euclidean distance between two points, identified by the subscript following it. δ is defined as

$$\delta(i,k;j,l) = \begin{cases} e^{-\frac{r}{q}} & r < q \\ 0 & \text{else} \end{cases}, \tag{2.11}$$

for some *q* for admissible matches. A match is not admissible when the corresponding features are further than a certain distance from each other, or their NCC is below a certain threshold. For an inadmissible match (*k,l*), δ(*i,kj,l*) equals 0.

Neighborhood constraint (NC), first proposed in [34], provides a similarity metric that takes both the intensity similarity and local structure, i.e., the layout of neighboring features around the pair, into consideration. Seemingly complex, it actually associates the features in *N(i)* and *N(j)* by using the similarity measure appearing as the operand in *max* operation in Equation 2.10, and weights the NCC similarity of features *i* and *j* with the total similarity of the matches found in their neighborhood. However, this "internal" association is not one-to-one, and does not accommodate for no-match cases. These problems are mitigated by utilizing the *auction algorithm* [67] that is described in the next section.

NC is a robust technique, especially when employed in a *guided matching* scheme, a method that is discussed in Section 2.2.3. However, it is computationally expensive, and in video sequences, NCC, a much simpler metric, performs almost as well as NC.

2.2.2 Association

The association of features with respect to their similarity scores is handled by the *auction algorithm* [67]. The auction algorithm is a well-known tool in the target tracking literature [84]. The algorithm simulates an auction session, in which the buyers have varying interest to the auctioned items. The measure of their interest is the amount they are willing to pay. The prices of the items change during the auction process, and these changes modify the interest levels of the buyers. In a feature association problem, features in one of the sets are the buyers, and in the other set are the items. Inadmissible matches are indicated with a similarity score of -∞. In order to allow no-match cases, the items are extended by the addition of "dummy" objects, which, in auction terminology, reflect the tendency not to spend money.

An auction session starts with all items at the price, P_j, of zero. Then, at each pass, a buyer without an item is chosen, and the item that interests him most is assigned to him, with interest defined as

$$\theta_{ij} = s_{ij} - P_j, \tag{2.13}$$

where s_{ij} represents the similarity between feature i and j, and P_j, the price of the item. If the item already has an owner, the owner is dispossessed. Finally, the price is updated as

$$P_j^{new} = P_j^{old} + y + \varepsilon, \tag{2.14}$$

where y is the difference between the maximum and the second greatest similarity scores for a buyer, i.e., the urge to buy that specific item, and ε is the minimum

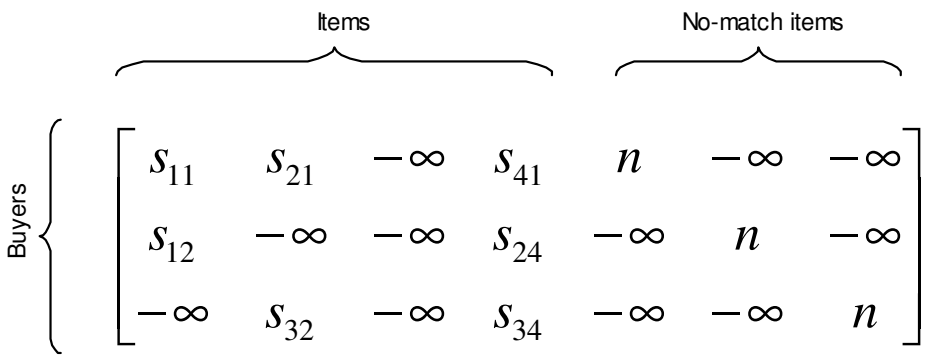


Figure 2.3: Similarity matrix for auction. *n* represents the similarity score for dummy items.

bid. The passes continue until every buyer is associated with an item, or the maximum number of iterations is reached, in which case, the remaining buyers are associated with the objects they want most. The algorithm attempts to maximize the sum of similarity scores for all assignments, and is guaranteed to yield a solution within ε of the maximum, unless terminated prematurely. In [84], a good rule of thumb for ε is stated as

$$\varepsilon = \frac{0.001}{n_i + n_b}, \tag{2.15}$$

where n_i and n_b represent the number of items and the numbers of buyers, respectively.

The algorithm is summarized below [67].

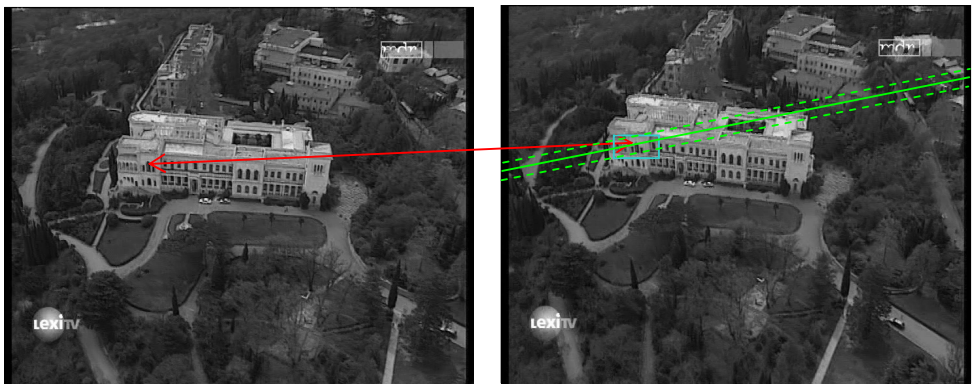


Figure 2.4: Guided matching. Red line indicates a corresponding feature pair. Blue rectangle is the search region suggested by the homography belonging to the *façade* of the building. Solid green line corresponds to the epipolar line, and the dashed lines delineate the search region.

`Algorithm: Auction`

Input: A matrix keeping the similarity scores for all admissible matches (see Figure 2.3).

Output: List of associated buyers and items, and unassociated items

1. Set all prices to zero.
2. While there exist any buyer without an item
    - ***a.*** Select a buyer without an item.
    - ***b.*** Give the buyer the item that interests it most.
    - ***c.*** Dispossess the current owner.
    - ***d.*** Update the price of the item and the interest level of the buyers towards the item.

When setting up an auction, limiting the admissible matches to a neighborhood of the feature (search region), and imposing a minimum similarity threshold were observed to prevent incorrect associations and to speed up the process.

2.2.3 Guided Matching

If an estimate of the projective mapping relating the feature sets is available, it can be used to constrain the search region for admissible matches. This mapping could be a homography, or a fundamental matrix (F-matrix). The former maps a point to another point, whereas the latter maps a point to a line (epipolar line) in the other image [10]. Then, a search region around the transferred point, or along the line can be determined, whose area depends on the accuracy of the estimate of the mapping. These concepts are illustrated in Figure 2.4.

When this mapping is not available, it can be estimated by using the matches obtained by conventional methods, and this estimate can be used to increase both the size and the quality of the correspondence set. In turn, the new correspondence set yields a better estimate of the mapping.

F-matrix, homography and the methods to estimate these entities are treated in detail in Chapters 3 and 4.

2.3 Feature Tracking

Two essential requirements for a successful 3D reconstruction are reliable feature pairs and sufficiently dissimilar camera poses (reasonably wide baseline and angle between the rays emanating from the camera centers and passing through the features). However, since the performance of matching algorithms deteriorates rapidly as the similarity between camera poses decrease, these two requirements are seemingly at conflict. A common solution to this problem is, instead of attempting to establish correspondences between two frames directly, tracking the features throughout the frames in between. This approach solves the matching

problem as in a similar pose setting, and provides feature matches for a wide-baseline reconstruction problem.

Features can be tracked as easily as by linking the matches established in successive frames of a sequence. The use of Kalman filter allows more sophisticated models and the incorporation of more information to the tracking process, such as displacement estimates obtained via KLT tracker. Both of these tools are introduced in the next section.

2.3.1 Discrete Kalman Filter

Kalman filter is a well-known tool for state estimation of dynamic systems [68]. The estimate is computed by using all available measurements, and Kalman filter is capable of incorporating an impressive amount of information about the process and the noise into the estimation procedure. This information is encapsulated in the state and measurement equations. The former defines the evolution of the system state, and the latter describes the relation between the state and the observations. The filter first propagates the current state estimate and its covariance according to the state equation, then updates the Kalman gain, computes the error between the actual and the predicted measurements from the observation equation, and finally scales this error with the Kalman gain to update the state estimate with measurements. This operation is algebraically formulated as [68]

$$\textit{State equation: } \mathbf{s}_k = \mathbf{A}_k \mathbf{s}_{k-1} + \mathbf{w}_k \tag{2.16}$$

$$\textit{Observation equation: } \mathbf{t}_k = \mathbf{O}_k \mathbf{s}_k + \mathbf{v}_k \tag{2.17}$$

$$
\begin{aligned}
&\hat{\mathbf{s}}_k = \mathbf{A}_{k-1}\hat{\mathbf{s}}_{k-1|k-1} \\
&\mathbf{P}_{k|k-1} = \mathbf{A}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{A}_{k-1}^T + \mathbf{Q}_k^w \\
\textit{Kalman filter equations:}\quad &\mathbf{G}_k = \mathbf{P}_{k|k-1}\mathbf{O}_k^T\left[\mathbf{O}_k\mathbf{P}_{k|k-1}\mathbf{O}_k^T + \mathbf{Q}_k^v\right]^{-1} \\
&\hat{\mathbf{s}}_{k|k} = \hat{\mathbf{s}}_{k|k-1} + \mathbf{G}_k\left[\mathbf{t}_k - \mathbf{O}_k\hat{\mathbf{s}}_{k|k-1}\right] \\
&\mathbf{P}_{k|k} = \left[\mathbf{I} - \mathbf{G}_k\mathbf{O}_k\right]\mathbf{P}_{k|k-1},
\end{aligned}
\tag{2.18}
$$

where **s** and **t** stand for the state and measurement vectors, **O** and **A** are the observation and the state matrices, **w** process noise, **v** observation noise, $\mathbf{Q}^w$ process noise covariance matrix, $\mathbf{Q}^v$ observation noise covariance matrix, **G** Kalman gain and **P** error covariance matrix. ^ denotes the estimate of the relevant quantity. All above entities are time varying, as suggested by the presence of the subscript *k*.

When the state, observation and covariance matrices are constant, a closely related recursive estimator is the fading memory filter. Using the previously established notation, it can be expressed as

$$
\hat{\mathbf{s}}_k = (1-\alpha)\hat{\mathbf{s}}_{k-1} + \alpha\mathbf{t}_k, \tag{2.19}
$$

where α is a constant weight.

2.3.2 Optical Flow Equation and Kanade-Lukas-Tomasi Tracker

Consider two images of a scene acquired by a moving camera with an interval of *t* units of time, and assume that the motion can be accurately modeled by a translation (a valid assumption for successive frames of a video sequence at

typical frame rates). Under these assumptions, the optical flow equation can be expressed as [69]

$$\frac{\partial I(x,y,t)}{\partial x} d_V + \frac{\partial I(x,y,t)}{\partial y} d_H + \frac{\partial I(x,y,t)}{dt} = 0, \tag{2.20}$$

where I is the image of the scene acquired by one of the cameras, and d_h and d_v are the displacements in the horizontal and vertical directions.

KLT tracker [16] is a tool to solve the optical flow equation for the displacements at discrete points, assuming that the displacements are constant within a block around the point. It also implicitly solves the matching problem, and builds feature trajectories by translating a feature using the estimated displacements. The main operation in KLT is the solution of the following equation [9]:

$$\mathbf{C}\begin{bmatrix} d_H \\ d_V \end{bmatrix} = \begin{bmatrix} \sum_{i,j \in N} w(i,j) I_H(i,j) I_t(i,j) \\ \sum_{i,j \in N} w(i,j) I_V(i,j) I_t(i,j) \end{bmatrix}. \tag{2.21}$$

In this equation, **C** is defined as cornerness matrix in Section 2.1.1, with w chosen as uniform weighting and temporal gradient I_t as the difference of the two images. The equation amounts to estimating the best displacement that minimizes the SSD between a reference and a target patch in two different images, i.e., registering the patches.

In [9], a pyramidal KLT implementation is proposed to improve the accuracy of the displacement estimates for relatively large displacements. This method solves the optical flow equation iteratively and across multiple resolutions (i.e., downsampled versions of the image). At each iteration, the reference patch is shifted towards the target patch with the displacement estimate from the previous

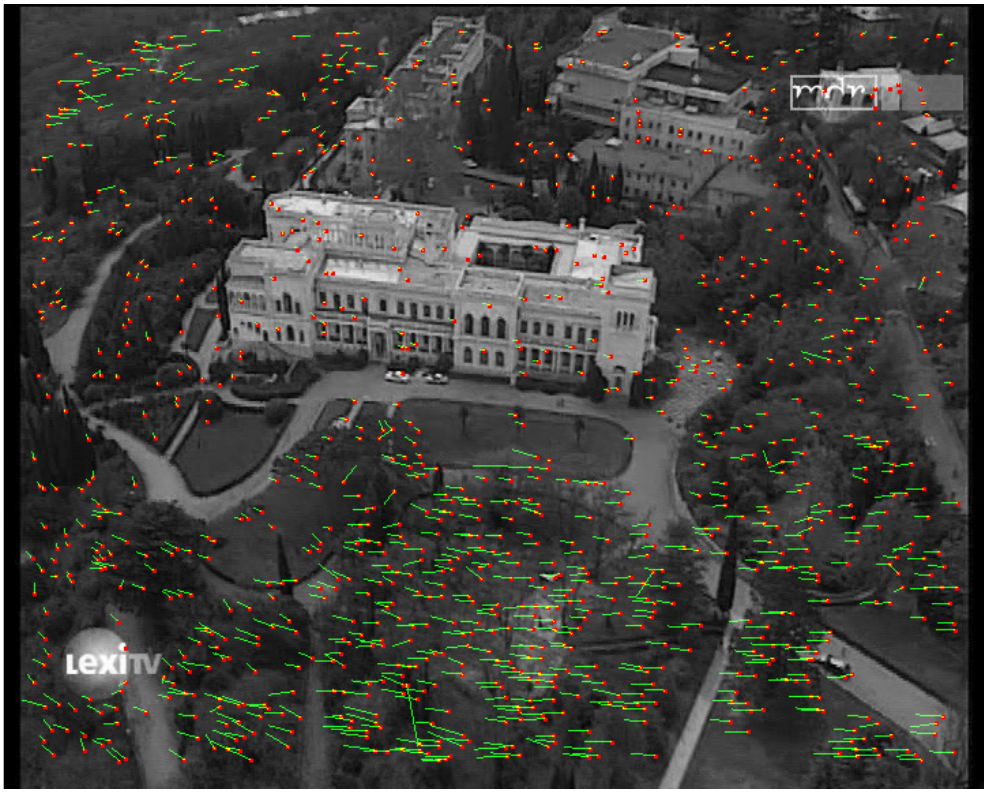


Figure 2.5: Feature displacements estimated by KLT. Red squares indicate the features, and green lines, the displacements. Displacement magnitudes are scaled by 4 to enhance visibility.

iteration. Once the iterations converge at a certain resolution, the final total displacement estimate at the current level is scaled up, and used as the initial estimate for the next resolution level. Figure 2.5 is a typical output of the algorithm.

The performance of KLT is determined by a number of parameters. The down-sampling rate and the number of levels control the range and accuracy of the displacement estimates. Typically, employing 3 levels, with a down-sampling rate of 2 at each, yields successful performance. For anti-aliasing, a suitable low-pass filter should be used. Two other parameters, the threshold on the determinant and the condition number of **C**, ensure that at each stage of the process, the patch is centered at a strong feature, and optical flow equation can be solved reliably.

The pyramidal KLT algorithm is presented below [9].

`Algorithm: Pyramidal Kanade-Lukas-Tomasi Tracker`

Input: Two frames, a feature list belonging to one of the frames, optionally, an initial displacement estimate for each feature.

Output: The displacement estimate at each feature.

1. Construct the image pyramid.
2. For each feature
    - ***a.*** Solve Equation 2.21 at the coarsest resolution, if **C** fulfills the determinant and condition number criterion. Otherwise, discard the feature.
    - ***b.*** Shift the patch, and repeat Step 2a until convergence.
    - ***c.*** Scale up the displacement and use it as initial estimate for the next resolution level.
    - ***d.*** Repeat Steps 2a-c for all levels.

2.4 Optical Flow-Based Tracking

Optical flow-based (OF) tracker is an improvement over basic KLT tracker, with the addition of a Kalman filter, and track initiation and maintenance mechanisms. OF tracker constructs a feature trajectory as a sequence of inter-frame displacements (translations), starting from a corner feature.

2.4.1 System Model

In OF tracker, a feature is defined by a state vector comprised of vertical and horizontal components of the feature position and displacement. The feature is assumed to be translating at a constant velocity, and velocity is the only available observation. Inverse of the **C** matrix is used as a measure of reliability, i.e., an approximation to the measurement covariance [93]. Model errors are represented by an independent Gaussian noise process, and similarly, observations are corrupted by Gaussian noise. In order to estimate the unknown state, a Kalman filter with the following specifications is used:

$$\mathbf{A}=\begin{bmatrix}1&0&1&0\\0&1&0&1\\0&0&1&0\\0&0&0&1\end{bmatrix} \qquad \mathbf{s}=\begin{bmatrix}c_V\\c_H\\d_V\\d_H\end{bmatrix}$$
$$\mathbf{O}=\begin{bmatrix}0&0&1&0\\0&0&0&1\end{bmatrix} \qquad \mathbf{t}=\begin{bmatrix}\hat{d}_V\\\hat{d}_H\end{bmatrix}. \tag{2.21}$$

In the above expression, c_V and c_H represent vertical and horizontal coordinates of the location, and, d_V and d_H, vertical and horizontal coordinates of the displacement, respectively.

2.4.2 OF Tracker

The basic operation of the tracker is simply extracting the displacement measurements for each feature, at the positions indicated by the Kalman filter, and feeding these measurements into the track maintenance module to update the trajectories. The tracking process starts with the extraction of initial features by Harris corner detector, and the estimation of initial displacements by KLT, to initialize the Kalman filters. Then, in each cycle, at the estimated positions of the features, the optical flow equations are solved to obtain the displacement measurements. The track maintenance module updates the tracks with measurements. However, it should be noted that it is not possible to obtain a reliable displacement estimate for each feature for a variety of reasons including occlusion, feature leaving the field-of-view, or an incorrect prediction of the feature position. In this case, the track maintenance module performs a no-measurement update, by propagating the state of the Kalman filter associated with the track, hoping that a new measurement will be received in the next frame. If the track receives no measurement for a number of consecutive turns or the ratio of no-measurement updates make up more than a certain percentage the trajectory length, the trajectory is declared as lost, and deleted. In order to capture the new scene features entering the field of view, at each frame, Harris corner detector extracts new corners in the regions not in the immediate vicinity of any of the existing tracks. These new corners are then used to initiate new tracks by the track maintenance module.

The tracker algorithm is depicted in Figure 2.6, and a typical output is presented in Figure 2.7. It can be summarized as follows:

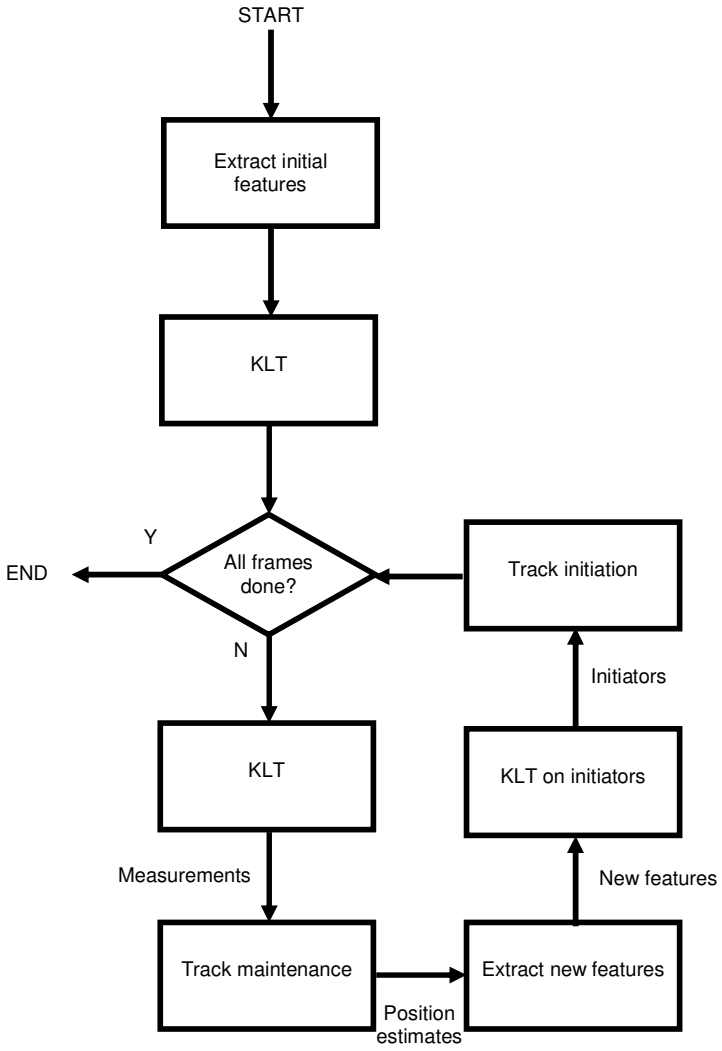


Figure 2.6: Flowchart of OF tracker.

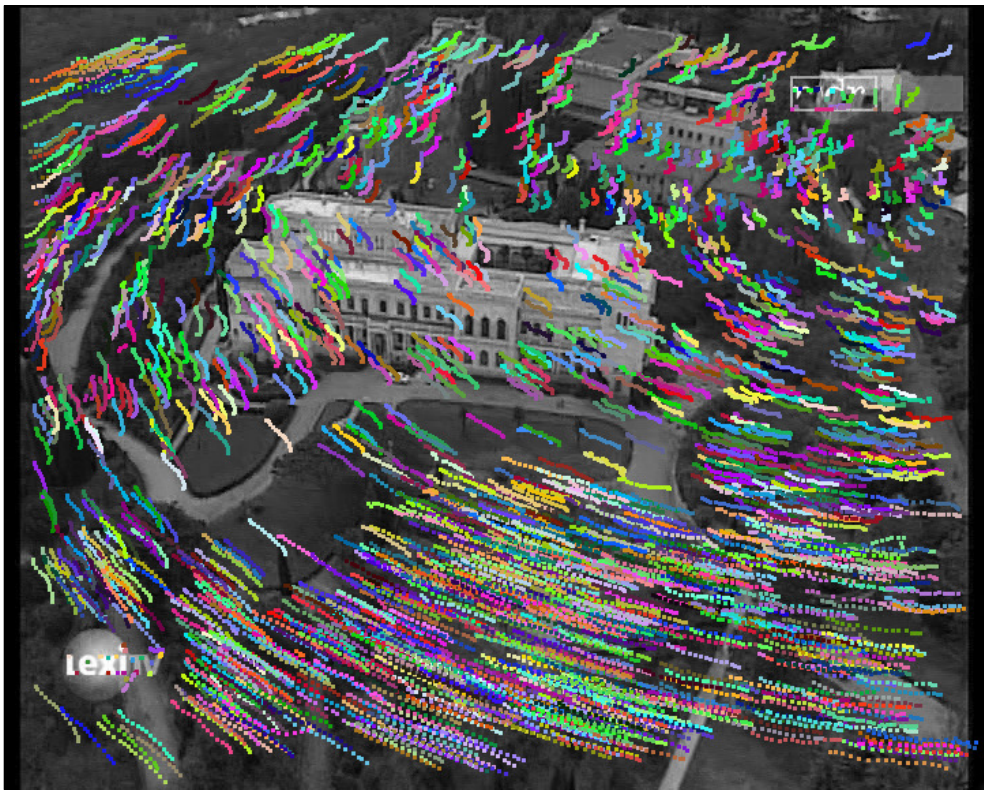


Figure 2.7: Results for OF tracking. Each color indicates a feature trajectory, and each square, a point on the trajectory of that color.

Algorithm: OF Tracker

Input: A frame sequence.

Output: Feature trajectories.

1. Construct the initial track set.
2. For all frames
    - ***a.*** Get the current feature position estimates.
    - ***b.*** Compute displacement measurements by KLT, using the current displacement estimates to initialize KLT.
    - ***c.*** Update tracks and delete lost tracks.
    - ***d.*** Extract new features and compute their initial displacement estimates by KLT.
    - ***e.*** Initiate the corresponding tracks.

2.5 Corner-to-Corner Tracking

Corner-to-corner (CC) tracker links individual matching results for the consecutive frames of a sequence to build trajectories. The major difference between the CC and OF trackers is the existence of an association block to solve the matching problem explicitly in CC tracker, instead of an implicit solution by KLT.

2.5.1 System Model

In CC tracker, the system state is composed of two sets of parameters. The first set corresponds to the displacement and position of the feature. As in OF tracker, constant velocity model is adopted. However, the measurement vector contains not only the displacement, but also the position of the feature. Again, $\mathbf{C}^{-1}$ is used

as an approximation to the measurement covariance. The Kalman filter corresponding to this model is given as

$$\mathbf{A}=\begin{bmatrix}1 & 0 & 1 & 0\\ 0 & 1 & 0 & 1\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad \mathbf{s}=\begin{bmatrix}c_V\\ c_H\\ d_V\\ d_H\end{bmatrix}$$
$$\mathbf{O}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad \mathbf{t}=\begin{bmatrix}\hat{c}_V\\ \hat{c}_H\\ \hat{d}_V\\ \hat{d}_H\end{bmatrix}. \tag{2.22}$$

The second set is comprised of the intensity values of a patch centered at the feature, or *context*. The context of a feature is assumed to be constant and the intensity value at each pixel is assumed to be independent of the others, and contaminated with constant-variance noise. Under these assumptions, it is possible to set up a Kalman filter. However, considering that all elements of the state are constant, fading memory filter is an equally capable alternative with far less computational and memory requirements. The context measurements are obtained from the image patches centered at the position measurements.

2.5.2 CC Tracker

CC tracker is similar to OF tracker in many respects, such as initialization and track maintenance. The main difference between these two trackers is in the tracking cycle. At each frame, CC tracker seeks measurements in the new frame to associate with the tracks. For each track, admissible measurements can only come from the corresponding search region, which is centered at the predicted location of the tracked feature in the new frame. Since the state vector describes the

tracked feature *after* the previous frame is processed, i.e., is an estimate of the state computed from all frames up to and including the previous frame, its location in the new frame is predicted by translating it to the new frame by the track velocity estimate kept in the state vector. Prior to translation, the velocity estimate is refined by incorporating the information present in the previous and the new frame, via KLT.

In order to solve the cases where there is more than one admissible measurement for a track, or a measurement is admissible for multiple tracks, association algorithm is employed. The similarity metric between a track and a measurement is defined as

$$S(i,j) = \frac{w_i E\{(I_i - \mu_i)(I_j - \mu_j)\}}{\sigma_i \sigma_j} + \frac{w_p}{1 + p_{ij}} + \frac{w_d}{1 + d_{ij}}, \tag{2.23}$$

where the first term is the normalized cross correlation defined in Section 2.2.1, p_{ij} and d_{ij} is the difference between the position and displacement of the state and the measurement, respectively, and w terms stand for the weights.

Another major difference between the trackers is in the measurement extraction stage. CC tracker extracts position and context, in addition to displacement. Besides, when a measurement is admissible for association with a track, its position is further refined by the utilization of KLT to minimize the difference between the measured and the predicted context (i.e., the context belonging to the track), prior to association. This practice is justified as an assertion of the constant intensity assumption, and utilization of a cost function more relevant to this assumption than that of subpixel refinement in Harris corner detector, for subpixel feature localization.

The tracker algorithm is depicted in Figure 2.8. It can be summarized as follows:

`Algorithm: CC Tracker`

Input: A frame sequence.

Output: Feature trajectories.

1. Construct the initial track set.
2. For all frames
    - ***a.*** Extract measurements by Harris corner detector and KLT.
    - ***b.*** Refine measurements to subpixel resolution by KLT.
    - ***c.*** Associate tracks with measurements.
    - ***d.*** Update tracks and delete lost tracks.
    - ***e.*** Extract new features and compute their initial displacement estimates by KLT.
    - ***f.*** Initiate the corresponding tracks.

2.6 Experimental Results

In order to assess the performance of OF and CC tracking approaches, two variants of each algorithm, KLT, OF and CC with and without KLT (CC w/KLT and w-o/KLT, respectively) subpixel refinement were tested. The test was composed of 18 sequences of indoor and outdoor real sequences, synthetic sequences, and artificial sequences, i.e., sequences only with a foreground object with simple features, and a blank background. Sample images from the test set are presented in Figure 2.9.

Two experiments were conducted. In the first experiment, the performance of the tracker while tracking an intensity structure (i.e., a corner) was investigated. In the

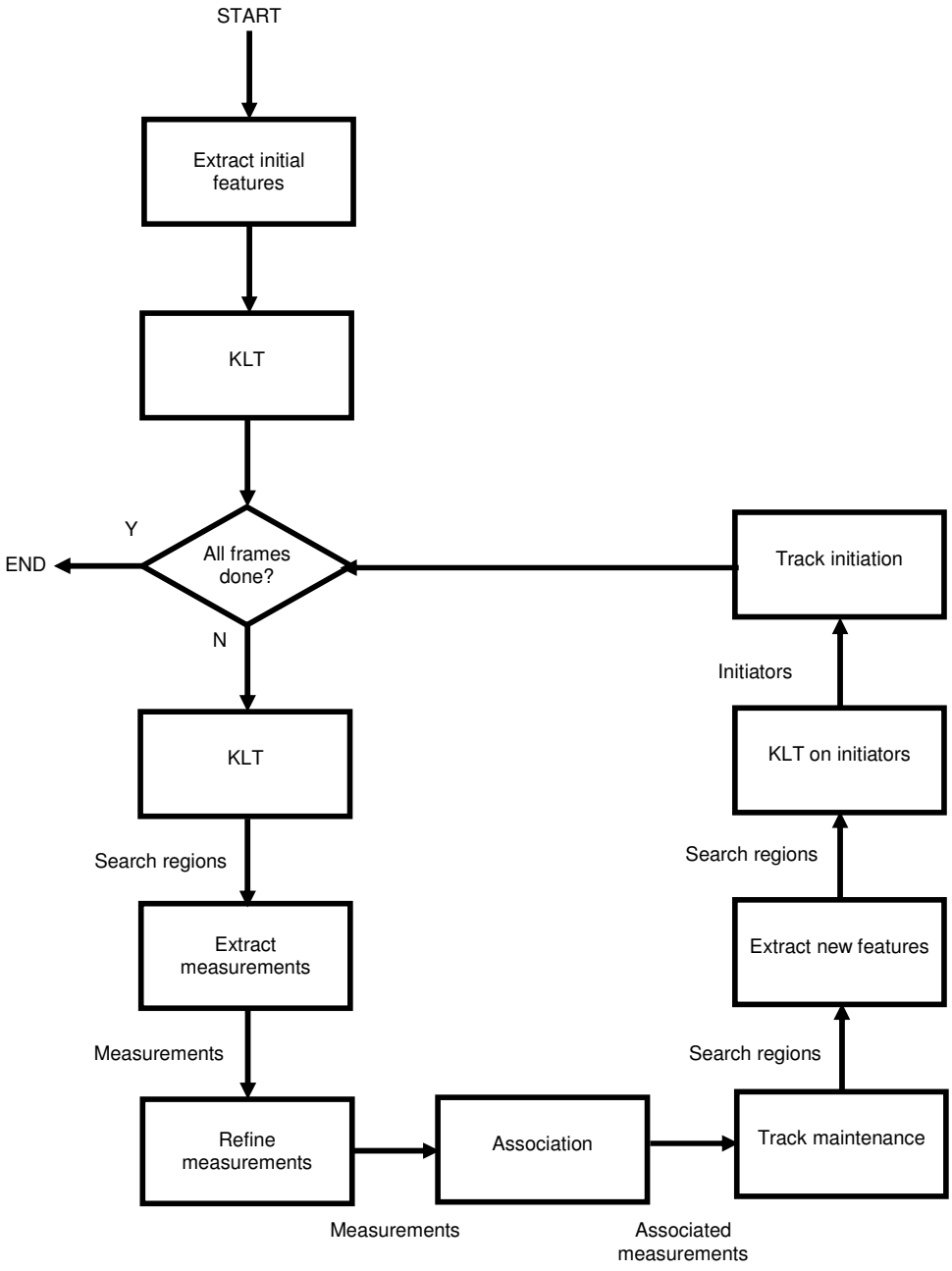


Figure 2.8: Flowchart of CC tracker.

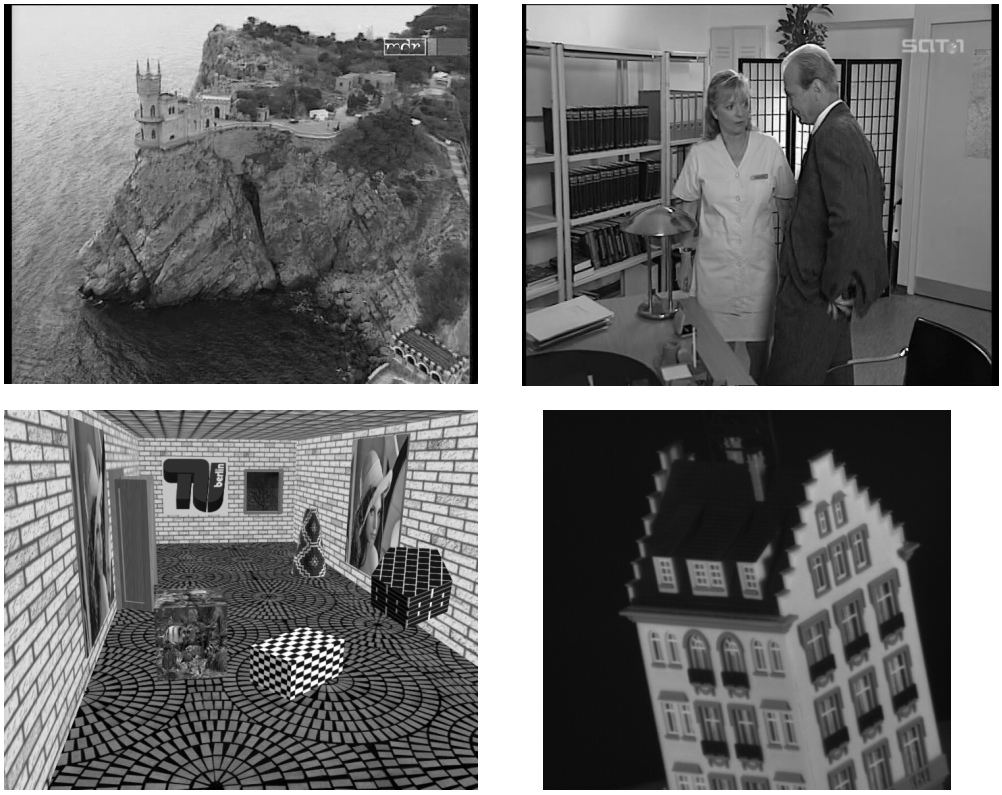


Figure 2.9: Sample images from the test set.

second experiment, the utility of the tracks generated by OF and CC w/KLT in F-matrix estimation was analyzed. In all experiments, the tracks with trajectories less than 15 frames were discarded, since the aim of tracking is to establish feature trajectories long enough to establish wide baseline correspondences.

2.6.1 Intensity Tracking

In this experiment, the aim was to evaluate the corner tracking performance of the trackers. The evaluation was performed by using the following criteria:

- **# Tracks:** Number of tracks.
- **Track length:** Mean track length.
- **Feasible/generated tracks (F/G):** Mean ratio of the tracks with trajectories longer than 15 frames, to all generated tracks.
- **Measurement/trajectory length (M/T):** Mean ratio of the measurements received by a trajectory to its total length, including both measurement and no-measurement updated points.
- **Residual MSE:** Median residual MSE of KLT.
- **Approximation of vertical and horizontal variances (VarV and VarH):** Median vertical and horizontal measurement variance approximations, i.e., the diagonal elements of $\mathbf{C}^{-1}$.
- **Prediction MSE:** Median prediction error of the Kalman filter

For each class of data, the experiment results were weighted with the sequence length.

The evaluation criteria for intensity tracking performance, in fact, have two different subgroups. #Tracks, track length and measurement ratio are related to the number of features tracked longer than 15 frames. Residual MSE and measurement variances are related to the reliability of the measurements. The experiment results that are presented in Tables 2.1-4 should be interpreted with this remark in mind.

Comparison between KLT and OF

In all data classes, basic KLT algorithm generated more tracks than OF, with a better M/T. However, while this result might seem to promote KLT, the considerable difference in track length and F/G indicates that KLT is inferior to OF in the ability to maintain long trajectories. The Kalman filter in OF improves the handling of no-measurement cases, thus makes it possible to establish longer trajectories. Longer trajectories, alone, are a reason to choose OF over KLT, as the aim of the tracker is to generate wide baseline matches. Besides, a large number of tracks implies that lost tracks are reinitiated under different identities. This causes multiple reconstructions for the same scene landmark. As for reliability, the optical flow equations are solved more accurately in OF, due to improved initial displacement estimates supplied by the Kalman filter. However, VarV and VarH are slightly better for KLT, as interpolated trajectory points are naturally less accurate than actual measurements, and as M/T indicates, KLT has a denser concentration of actual measurements.

Comparison between CC w/KLT and CC w-o/KLT

Comparison between KLT with OF clearly indicates the advantages of employing a Kalman filter. Therefore, in CC experiments, both variants are equipped with Kalman filters.

Table 2.1: Experiment results for intensity tracking in outdoor sequences. The best results are marked with red and bold.

Outdoor 8 sequences- 2425 frames	**KLT**	**OF**	**CC w/o KLT**	**CC w/KLT**
#Tracks	**5521**	3598	2615	2657
Length (frames)	33	**45**	35	35
Feasible/Generated	0.31	**0.34**	0.08	0.09
Measurement/Length	**0.97**	0.88	0.81	0.81
KLT Residual MSE (Eqn. 2.1)	25.98	**24.14**	N/A	26.63
VarV (1e-4) (pixels)	2.71	3.05	N/A	**2.40**
VarH (1e-4) (pixels)	3.20	3.54	N/A	**3.18**
Prediction MSE (Eqn. 2.18)	N/A	**0.15**	0.34	0.19

Table 2.2: Experiment results for intensity tracking in indoor sequences. The best results are marked with red and bold.

Indoor 5 sequences- 683 frames	**KLT**	**OF**	**CC w/o KLT**	**CC w/KLT**
#Tracks	**3417**	3045	2438	2689
Length (frames)	29	**42**	32	32
Feasible/Generated	0.22	**0.32**	0.08	0.08
Measurement/Length	**0.97**	0.86	0.81	0.81
KLT Residual MSE (Eqn. 2.1)	31.05	32.88	N/A	**30.90**
VarV (1e-4) (pixels)	1.46	1.65	N/A	**1.41**
VarH (1e-4) (pixels)	**1.51**	1.70	N/A	1.52
Prediction MSE (Eqn. 2.18)	N/A	**0.07**	0.20	0.09

Table 2.3: Experiment results for intensity tracking in artificial sequences. Best results are marked with red and bold.

Artificial 4 sequences- 376 frames	**KLT**	**OF**	**CC w/o KLT**	**CC w/KLT**
#Tracks	**461**	374	373	421
Length (frames)	50	**76**	59	54
Feasible/Generated	0.42	**0.54**	0.12	0.14
Measurement/Length	**0.99**	0.98	0.90	0.90
KLT Residual MSE (Eqn. 2.1)	15.61	15.16	N/A	**14.51**
VarV (1e-4) (pixels)	2.11	2.20	N/A	**2.10**
VarH (1e-4) (pixels)	2.04	2.10	N/A	**1.86**
Prediction MSE (Eqn. 2.18)	N/A	**0.02**	0.07	0.02

Table 2.4: Experiment results for intensity tracking in synthetic sequences. Best results are marked with red and bold.

Synthetic 3 sequences- 653 frames	**KLT**	**OF**	**CC w/o KLT**	**CC w/KLT**
#Tracks	**10993**	7587	7289	7821
Length (frames)	26	33	**36**	34
Feasible/Generated	0.180	**0.22**	0.08	0.09
Measurement/Length	**0.96**	0.83	0.83	0.82
KLT Residual MSE (Eqn. 2.1)	16.94	15.82	N/A	**13.51**
VarV (1e-4) (pixels)	**0.58**	0.61	N/A	0.59
VarH (1e-4) (pixels)	**0.80**	0.91	N/A	0.92
Prediction MSE (Eqn. 2.18)	N/A	**0.01**	0.09	0.02

In terms of quantity, both trackers have a similar performance, with CC w/KLT having a slight edge in F/G and CC w-o/KLT in trajectory length, in non-real sequences. However, the predictions of the Kalman filter are invariably better when KLT is employed, implying superiority in the quality of the trajectories.

Comparison between OF and CC w/KLT

The experiments above imply that OF and CC w/KLT outperform their simpler variants. However, when it comes to a relative evaluation for OF and CC in terms of their intensity tracking performance, the result is less clear, as the experiments indicate that OF offers longer and, in real sequences, more trajectories, while CC generates higher quality ones.

The reason that the CC tracker offers better quality is clear: It utilizes more information, and in effect a much more elaborate measurement extraction and association scheme. As for the quantity advantage of the OF tracker, there are several causes. First, the corner detector allows only a single corner in a certain neighborhood. Therefore, when two tracks are close enough, their search regions overlap, and only one of them can get a new measurement, causing the other to starve and die. However, in OF tracker, there is no such restriction.

Another cause is the fact the NCC is only translation-invariant. The association stage in CC involves an NCC thresholding, and even slight rotations, coupled with noise, may yield a very low NCC, thus, eliminate the candidate. The correction of measurements by KLT alleviates this problem only to a certain extent, as the predicted context is a weighted average over time; and hence smoothed, degrading the performance of KLT. The OF tracker has an analogous threshold, the convergence threshold of KLT. However, iterative registration performed by KLT allows a better NCC score, and it is possible that the threshold used in KLT actually corresponds to a lower NCC threshold than that of the association module in CC tracker. However, since the relation is not straightforward, such a claim is

hard to confirm. F/G seems to lend support to this claim, but the gap between the two trackers probably stems from the choice of a lower cornerness threshold for the corner detector in CC tracker, to encourage generation of a comparable number of tracks with the OF tracker.

One final possible cause is inspired by the comparison of the performances in real and synthetic sequences. Synthetic sequences are generated with the ease of tracking in mind, i.e., the sequence is noise-free, and the features are sharp and relatively sparse to prevent false associations. Therefore, both trackers track the same features with a similar performance. However, in real data none of the above conveniences exist, and such data forces the CC tracker to cope with false matches and smooth corners. In contrast, pyramidal KLT, the core component of the OF tracker, retains its reliability in such cases.

2.6.2 Fundamental Matrix Estimation

In this experiment, the utility of the trajectories for F-matrix estimation purposes was evaluated. To this aim, a set of frame pairs yielding a relatively reliable F-matrix estimate was identified via GRIC and then F-matrices are constructed and outliers are eliminated. The evaluation is based on the following criteria

- **Average Sampson error:** Median average Sampson error per inlier.
- **Inlier ratio:** Mean ratio of inliers to all pairs.

F-matrix and outlier elimination are detailed in Section 3.2. The definitions of Sampson error and GRIC are presented in Equations 3.11 and 3.12, respectively.

The experiment results, presented in Table 2.5, indicate that OF outperforms CC slightly, but almost invariably, in all data classes. This implies that OF generates

Table 2.5: Experiment results for F-matrix estimation. Best results are indicated with red and bold

OF Tracker	Real Indoor	Real Outdoor	Synthetic	Artificial
Sampson/Inlier (pixels) (Equation 3.11)	**0.03**	**0.03**	0.02	**0.00**
Inlier/Total	**0.89**	**0.95**	**0.92**	**0.94**

CC Tracker	Real Indoor	Real Outdoor	Synthetic	Artificial
Sampson/Inlier(pixels) (Equation 3.11)	0.04	0.04	**0.02**	0.01
Inlier/Total	0.85	0.90	0.89	0.90

sufficiently accurate trajectories, so that the quantity advantage of OF surpasses the quality advantage of CC, as longer trajectories and greater number of correspondences facilitate the estimation of F-matrix. The observation that the performance gap is narrowest in the synthetic case, in which both trackers use a similar amount of data, lends support to this conclusion.

2.7 Conclusion

In this chapter, fundamental building blocks of a feature tracker, namely, feature extraction, matching and tracking are introduced. Then, two feature tracking approaches, corner-to-corner tracking and optical flow-based tracking are discussed. Two trackers, each adhering to one of these approaches are designed, and their performances are experimentally compared. The experiments indicate that OF tracker is superior to CC tracker in quantitative terms, while qualitatively, CC is better. However, in the F-matrix estimation experiments, which is more relevant to the ultimate aim of the tracker, generating trajectories for 3D reconstruction, OF tracker is observed to have a slight, but consistent advantage.

CHAPTER 3

SPARSE 3D SCENE RECONSTRUCTION FROM UNCALIBRATED 2D VIDEO SEQUENCES

Sparse 3D scene reconstruction involves the solution of MFSfM problem. Contrary to what the chapter title suggests, it is a vital part of all 2D-3D conversion systems not because of the sparse 3D point cloud, but due to the camera matrix estimates, as camera matrices are the only required input, along with the video sequence, for depth-map based representations. Nevertheless, when, as in this work, a mesh-based representation is pursued, an accurate sparse 3D structure estimate also becomes indispensable.

In this work, MFSfM problem is solved by *prioritized sequential 3D reconstruction* algorithm [82], one of the main contributions of this thesis. The algorithm has 3 pillars. The first one is the projective two-view reconstruction, which also serves as an illustrative case-study for the introduction of basic tools, such as F-matrix estimation and triangulation. The second one is the extension of this algorithm to a sequential MFSfM technique, by employing the 2D-3D correspondences for the camera matrix estimates. The final pillar is the *prioritization* scheme, which seeks to establish a processing order for all the available frame pairs, to avoid from numerically unstable cases, and to facilitate convergence to a good solution. These topics are the main focus of this chapter,

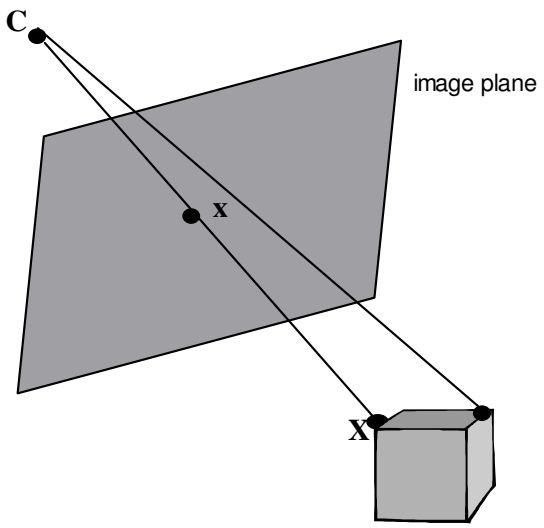


Figure 3.1: Image formation using pinhole camera model.

and are discussed in the following sections. Besides, two sections are devoted to self-calibration and metric reconstruction, for the cases when the required output ambiguity is at metric level, and one section is dedicated to the segmentation of independently moving objects, for a discussion of how to handle dynamic scenes, i.e., scenes with multiple motions.

3.1 Image Formation and Reconstruction Ambiguity

The algebra of 3D reconstruction is expressed exclusively via *projective geometry*. In this domain of mathematics, all entities are represented homogeneously. The homogeneous representation is an equivalence class in which the equivalence relation, $\mathbf{A} \equiv c\mathbf{A}$ **,** holds true, when **A** is an algebraic entity, and c is a non-zero real number. In Chapter 3 and 4, the homogeneous representation is used throughout, unless otherwise stated.

As illustrated in Figure 3.1, under the *pinhole camera* model, image formation is a 3D-to-2D mapping, defined by the camera center (**C** in the figure) and the orientation of the image plane, i.e., the surface on which the image is formed [10]. The relation is algebraically expressed as

$$\mathbf{x} \approx \mathbf{PX}, \tag{3.1}$$

where **X** and **x** are 4x1 and 3x1 vectors denoting homogeneous coordinates of a 3D point and its projection, respectively, and **P**, is the mapping, a 3x4 matrix known as *camera* or *projection matrix*. The symbol ≈ indicates projective equivalence.

SfM techniques use the projections of **X** in different views to solve for the structure and camera parameters. However, a certain correspondence set does not yield a unique (**P**;**X**) pair. An arbitrary transformation of coordinate system by **H** transforms the camera matrix and structure to ($\mathbf{PH};\mathbf{H}^{-1}\mathbf{X}$), a pair still creating exactly the same image features, as

$$\begin{aligned} \mathbf{x} &\approx (\mathbf{PH})\left(\mathbf{H}^{-1}\mathbf{X}\right) \\ &\approx \mathbf{PX}. \end{aligned} \tag{3.2}$$

This relation gives rise to an ambiguity that cannot be resolved by using the feature correspondences alone. In this dissertation, two types of ambiguities are encountered:

- **Projective ambiguity: P** and **H** are arbitrary homogeneous matrices of rank 3 and 4, respectively. Only a limited number of geometric properties

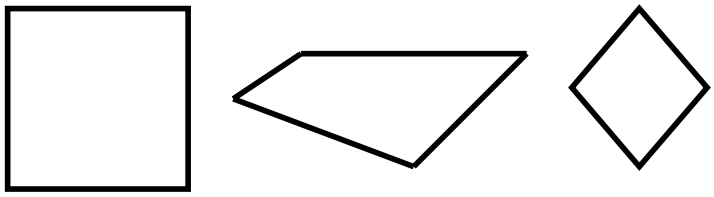


Figure 3.2: Reconstruction ambiguity in 2D. *Left:* Original shape. *Center:* A projective equivalent. *Right:* A metric equivalent

are invariant under projective transformations, such as concurrency, colinearity and order of contact.

- **Metric ambiguity: P** has to satisfy certain algebraic properties (See Section 3.3), and these additional constraints limit the choice of possible **H** matrices to similarity transformations, that can modify only scale, rotation and translation. The forms of the objects are invariant under similarities.

Figure 3.2 graphically illustrates examples of a projective and a metric distortion introduced by these ambiguities.

3.2 Projective 3D Reconstruction from Two-Views

Projective reconstruction from two views is a classical problem in SfM with a well-established solution [10], involving an alternating scheme. First, the corresponding feature pairs are used to estimate an algebraic entity, F-matrix that best explains the observed correspondences. This step amounts to locating the cameras. Then, the feature pairs are moved to the closest locations that satisfy the epipolar constraint imposed by this F-matrix. Finally, the camera matrices and the "corrected" feature positions are used to estimate a projective 3D structure by the method of triangulation [10].

3.2.1 Estimation of Fundamental Matrix

Fundamental Matrix

The relation between the two views of a scene is established by the epipolar geometry of these views. Epipolar geometry exists as long as the camera centers of the views do not coincide, and it can be uniquely determined unless the scene points are in a degenerate configuration, such as a plane. F-matrix is the entity that represents the epipolar geometry algebraically.

F-matrix has the following properties [10]:

- F-matrix is independent of structure. It is determined solely by the relative pose and internal parameters of the cameras.
- F-matrix is a 3x3 matrix. It is rank 2, and has 7 degrees of freedom.
- **Epipole:** Right and left null vectors of F-matrix correspond to the projections of the camera centers to the image plane. These points are called *epipoles*.
- **Epipolar lines:** F-matrix maps the points in one of the images to lines in the other. This mapping is expressed as

$$\begin{aligned} \mathbf{l}' &= \mathbf{F}\mathbf{x} \\ \mathbf{l} &= \mathbf{F}^{\mathbf{T}}\mathbf{x}', \end{aligned} \tag{3.3}$$

  where **(x;x')** is a corresponding feature pair.

- **Epipolar constraint:** F-matrix imposes the following constraint on the correspondences:

$$\mathbf{x}'^{\mathbf{T}}\mathbf{F}\mathbf{x} = \mathbf{0}, \tag{3.4}$$

- Given two camera matrices **P** and **P'**, the corresponding F-matrix equals to [10]

$$\mathbf{F} = \left[\mathbf{e}'\right]_{\times}\mathbf{P}'\mathbf{P}^{+}, \tag{3.5}$$

where $\mathbf{P}^{+}$ is the pseudo-inverse of $\mathbf{P}$ and the following relations hold [10]:

$$\begin{aligned} \mathbf{e}' &= \mathbf{P}'\mathbf{C} \\ \mathbf{P}\mathbf{C} &= \mathbf{0}. \end{aligned} \tag{3.6}$$

That is, $\mathbf{e}'$ is the projection of the camera center of the first camera, $\mathbf{C}$ to the second image. $[\mathbf{e}]_{\times}$ is defined as

$$\left[\mathbf{e}\right]_{\times} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}_{\times} = \begin{bmatrix} 0 & -e_3 & e_2 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{bmatrix}. \tag{3.7}$$

- An F-matrix defines a canonical camera pair (**P;P'**) as

$$\begin{aligned} \mathbf{P} &= \left[\mathbf{I} \mid \mathbf{0}\right] \\ \mathbf{P}' &= \left[\left[\mathbf{e}'\right]_{\times}\mathbf{F} \mid \mathbf{e}'\right] \end{aligned} \tag{3.8}$$

Linear Estimation of Fundamental Matrix by 8-Point Algorithm

For a given correspondence pair, epipolar constraint can be expanded to yield

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0, \tag{3.9}$$

where x and y are the individual coordinates of $\mathbf{x}$, and x' and y', that of $\mathbf{x}'$. f_{ij} are the elements of the F-matrix. Since F-matrix is a homogeneous entity, 8 of these equations can be used to set up a homogeneous equations system, whose null vector is the desired solution, up to a scale [31]. However, the result is not guaranteed to be rank-2. The rank constraint is enforced by setting its smallest singular value to 0. This yields a rank-2 matrix closest to the original result in the Frobenius-norm sense. If more than 8 correspondences are available, they can be

utilized by extending the equation system. The solution of this extended equation system is its right-singular vector belonging to the smallest singular value.

Normalized 8-point algorithm [12] is a significant improvement over the basic method. This approach improves the condition number; hence, the robustness of the equation system by normalizing the feature sets, so that the centroid of each feature set is at the origin of the coordinate system of its image and mean distance to the origin is $\sqrt{2}$. Once the normalized equation set is solved, the result is denormalized to find the F-matrix corresponding to the original features as follows [12]:

$$\mathbf{F} = \mathbf{T}'\mathbf{F}_\mathbf{N}\mathbf{T}. \tag{3.10}$$

In this relation, $\mathbf{F_N}$ is the normalized F-matrix, and $\mathbf{T}$ and $\mathbf{T}'$ are the transformations for each feature set.

Normalized 8-point algorithm is summarized below [12]:

Algorithm: Normalized 8-Point

Input: At least 8 feature correspondences.

Output: The F-matrix relating the frames.

1. Compute the normalizing transforms for features belonging to each frame and normalize the features.
2. Linearly solve for F-matrix by using Equation 3.9.
3. Compute the best rank-2 estimate of the result.
4. Denormalize the result by Equation 3.10.

Robust Estimation of F-Matrix by RANSAC

As mentioned above, when more than the minimum number of correspondences is available, normalized 8-point algorithm finds a least squares solution. However, this solution is not robust in the presence of outliers. A radically different approach is to use as few data as possible, assuming that this minimal data set is not contaminated by the outliers. RANSAC [70] offers a *hypothesize-and-test* framework for this approach. Hypothesizing involves generating an instance of the model from a randomly selected minimal set. For F-matrix estimation, when the model generator is designated as normalized 8-point algorithm, this minimal set is comprised of 8 points. This hypothesis is tested by evaluating the fitness of the estimated model to the available data, i.e., correspondences. Two popular metrics are the number of inliers, and GRIC [32]. The inliers are determined by the *Sampson distance* of the correspondences to the hypothesized F-matrix. The Sampson distance is defined as [10]

$$d_{Sampson} = \frac{\left(\mathbf{x}'^{\mathbf{T}}\mathbf{F}\mathbf{x}\right)}{l_x^2 + l_y^2 + l_x'^2 + l_y'^2}$$
$$\mathbf{l}' = \begin{bmatrix} l_x' \\ l_y' \\ l_z' \end{bmatrix} = \mathbf{F}\mathbf{x} \text{ and } \mathbf{l} = \mathbf{F}^{\mathbf{T}}\mathbf{x}', \tag{3.11}$$

where **x'** and **l'** denote the correspondence to a feature **x**, and the associated epipolar line, respectively, as defined in Section 3.2.1.

An outlier is a feature pair, whose distance to the model is above a certain threshold, determined by the noise on the coordinates of the features.

GRIC, defined below, is shown to be slightly superior to naïve inlier counting [34]:

$$GRIC = \sum_i \min\left(\frac{e_i}{\sigma^2}, \omega_3 (r-d) \right) + w_1 dn + w_2 k. \qquad (3.12)$$

In this expression, e_i is the fitness for the i^{th} pair to the model (F-matrix), σ, standard deviation of noise at each coordinate, d, the dimension of the model (3 for F-matrix), r, the dimension of data (4 for a 2D correspondence pair), k the degrees of freedom for the model, and n, total number of correspondence pairs. The parameters w_1, w_2 and w_3 stand for the weights of the individual terms. The fitness of a pair to a given F-matrix can be assessed through various methods including algebraic distance and Sampson distance [10].

RANSAC generates enough hypotheses to guarantee that at least one outlier-free sample is drawn from the data with a probability p. It can be shown that the number of necessary hypotheses equals [10]

$$N = \frac{\log(1-p)}{\log(1-\lambda^s)}, \qquad (3.13)$$

where λ is the ratio of inliers in the data set, and s is the size of the minimum sample set, 8 for F-matrix. λ can be adaptively determined as the ratio of inliers for the best model so far, to the total number of correspondences.

Totally random selection of data may occasionally produce sample sets composed of samples concentrated to a certain part of the image, thus, low representative

value. This is remedied by *bucketing,* i.e., dividing the frame into blocks, and drawing at most a single correspondence pair from a certain block for each sample set.

The output of RANSAC can be further improved by a final application of the normalized 8-point algorithm, as it is an F-matrix computed from only 8 equations. Use of an extended equation set constructed from the inliers definitely yields a more accurate estimate. This new F-matrix has another inlier set, which is used to compute another F-matrix. This procedure is iterated until the inlier set stabilizes or the evaluation score converges.

Robust F-matrix estimation with RANSAC is summarized below [10].

`Algorithm: F Estimation with RANSAC`

Input: Feature correspondences.

Output: F-matrix and a list of inliers.

1. While the number of generated hypotheses is below N
    - ***a.*** Draw 8 feature pairs from the data set randomly.
    - ***b.*** Use normalized 8-point algorithm to generate a hypothesis.
    - ***c.*** Test the hypothesis.
    - ***d.*** If the hypothesis is better than the current best hypothesis, save the current hypothesis, and update λ and N.
2. Until the evaluation score converges
    - ***a.*** Use all inliers from the previous iteration to estimate the F-matrix via normalized 8-point algorithm.
    - ***b.*** If the estimate is better than the current best estimate, update.

Robust Homography Estimation with RANSAC

A relevant topic is the estimation of homographies, that is, mappings of the type $\mathbf{x}' \approx \mathbf{Hx}$. A 2D homography relates the images of a plane in 3D. Estimation of a 2D homography is very similar to that of F-matrix. Normalized 8-point algorithm has a 4 point analogue for homographies, as each feature correspondence gives rise to the following two equations, expressed using the notation in Equation 3.9:

$$\begin{aligned} -xh_{21} - yh_{22} - h_{23} + y'xh_{31} + y'yh_{32} + y'h_{33} = 0 \\ xh_{11} + yh_{12} + h_{13} - x'xh_{31} - x'yh_{32} - x'h_{33} = 0. \end{aligned} \tag{3.14}$$

The application of RANSAC differs from the F-matrix case only in the definition of the Sampson error. Sampson error for a homography estimation is defined as [10]

$$d_{sampson} = \boldsymbol{\varepsilon}^{\mathbf{T}} \left(\mathbf{J}\mathbf{J}^{\mathbf{T}} \right)^{-\mathbf{1}} \boldsymbol{\varepsilon}, \tag{3.15}$$

where

$$\begin{aligned} \boldsymbol{\varepsilon} &= \begin{bmatrix} 0 & 0 & 0 & -x & -y & -1 & y'x & y'y & y' \\ x & y & 1 & 0 & 0 & 0 & -x'x & -x'y & -x' \end{bmatrix} \mathbf{h} \\ \mathbf{h} &= \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{bmatrix}^{\mathrm{T}} \end{aligned} \tag{3.16}$$

and

$$\mathbf{J}=\begin{bmatrix} -h_{21}+y'h_{31} & -h_{22}+y'h_{32} & 0 & xh_{31}+yh_{32}+h_{33} \\ h_{11}-x'h_{31} & h_{12}-x'h_{32} & -xh_{31}-yh_{32}-h_{33} & 0 \end{bmatrix}. \tag{3.17}$$

3.2.2 Triangulation

Figure 3.1 implies that given a camera matrix and a 2D point on its image plane, the 3D point projecting to it lies on the ray that emanates from the camera center and passing through the 2D point. If the same 3D point is imaged by a second camera, similarly, another ray can be traced from the second camera center. Therefore, the 3D point lies at the intersection of these rays. This process is called *triangulation*.

The most challenging aspect of triangulation is to find the 3D location of a point when the rays from its projections do not intersect, due to errors in the feature coordinates and camera parameters. *Optimal triangulation*, proposed in [36] aims to solve this problem by finding for each feature pair ($\mathbf{x}$; $\mathbf{x}'$), the pair ($\mathbf{x_c}$; $\mathbf{x}'_\mathbf{c}$) closest to it in the Euclidean sense, and subject to the constraint $\mathbf{x}'_\mathbf{c}\mathbf{F}\,\mathbf{x_c}=\mathbf{0}$. The rays passing through the new pair are guaranteed to intersect, and the solution is provably optimal in the *reprojection error* sense (See Section 3.6), if the noise on feature coordinates is Gaussian [10]. The solution involves the roots of a 6^{th} degree polynomial, whose construction requires an F-matrix in a special form. To this aim, first, the feature coordinates are translated to the origin by [36]

$$\mathbf{T}=\begin{bmatrix} 1 & 0 & -x \\ 0 & 1 & -y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}'=\begin{bmatrix} 1 & 0 & -x' \\ 0 & 1 & -y' \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.18}$$

where x and y are defined in Equation 3.9. Then, the epipoles are taken to x-axis via a rotation defined as [36]

$$\mathbf{R} = \begin{bmatrix} e_1 & e_2 & 0 \\ -e_2 & e_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{R}' = \begin{bmatrix} e_1' & e_2' & 0 \\ -e_2' & e_1' & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.19}$$

where

$$\mathbf{F_T} = \mathbf{T}'^{-\mathbf{T}} \mathbf{F} \mathbf{T}^{-\mathbf{1}} \tag{3.20}$$

and $\mathbf{e}$ and $\mathbf{e}'$ are right and left epipoles of $\mathbf{F_T}$, and e_i are as defined in Equation 3.7. The F-matrix in the desired form is obtained by applying this rotation to $\mathbf{F_T}$, i.e.,

$$\mathbf{F_R} = \mathbf{R}' \mathbf{F_T} \mathbf{R}^{\mathbf{T}}. \tag{3.21}$$

Following these transformations, the pencil of epipolar lines in each image can be parameterized by a single variable, t, as follows [36]:

$$\begin{aligned} \mathbf{l}(t) &= \begin{bmatrix} te_{R3} \\ 1 \\ -t \end{bmatrix} \\ \mathbf{l}'(t) &= \mathbf{F_R} \begin{bmatrix} 0 \\ t \\ 1 \end{bmatrix}. \end{aligned} \tag{3.22}$$

Since the features are mapped to the origin, their total distance to the corresponding epipolar lines becomes [36]

$$s(t) = \frac{t^2}{1 + e_{R3}^2 t^2} + \frac{(f_{R32}t + f_{R33})^2}{(f_{R22}t + f_{R23})^2 + e_{R3}^{'2}(f_{32}t + f_{33})^2}. \tag{3.23}$$

It should be noted that each value of t defines a candidate for the corrected feature pair in the transformed coordinate system, as the distance between a point and a line is the distance to the closest point on the line. Therefore, minimizing $s(t)$ is equivalent to finding a feature pair that satisfies the epipolar constraint, and closest to the features in the transformed coordinates. An inverse-transform yields the solution.

The minima of $s(t)$ can be computed by finding the roots of its derivative with respect to t, which is a 6^{th} degree polynomial. The real roots of the polynomial and ∞ are the candidates for the global minimum. This value, t_{min} is used to solve for the epipolar lines $\mathbf{l}$ and $\mathbf{l}'$. The closest points on these lines to the origins, ($\mathbf{x_{cR}}$; $\mathbf{x}'_{\mathbf{cR}}$), is the solution in the transformed coordinate system. Finally, these points are transferred back to the original coordinate system by

$$\begin{aligned} \mathbf{x_c} &= \mathbf{T}^{-1}\mathbf{R}^{\mathbf{T}}\mathbf{x_{cR}} \\ \mathbf{x}'_{\mathbf{c}} &= \mathbf{T}'^{-1}\mathbf{R}'^{\mathbf{T}}\mathbf{x}'_{\mathbf{cR}}. \end{aligned} \tag{3.24}$$

The 3D point corresponding to the corrected feature pair is the solution of the equation

$$\begin{bmatrix} x\mathbf{p}^3 - \mathbf{p}^1 \\ y\mathbf{p}^3 - \mathbf{p}^2 \\ x'\mathbf{p}'^3 - \mathbf{p}'^1 \\ y'\mathbf{p}'^3 - \mathbf{p}'^2 \end{bmatrix} \mathbf{X} = \mathbf{0}, \tag{3.25}$$

where $\mathbf{p}^i$ stands for the i^{th} row of the camera matrix **P**, and $\mathbf{p'}^i$, for **P'**.

The optimal triangulation algorithm is given below [36].

`Algorithm: Optimal Triangulation`

Input: Feature pairs and the F-matrix relating them.

Output: 3D point cloud.

1. For each feature pair
    - ***a.*** Transform the coordinate system so that the features are at the origin and the epipoles are on the x-axis.
    - ***b.*** Find the minimum of Equation 3.23.
    - ***c.*** Evaluate the epipolar lines.
    - ***d.*** For each line find the closest point to the origin.
    - ***e.*** Transform the points back to the original coordinate system.
    - ***f.*** Solve Equation 3.25 to find **X**, the corresponding 3D point.

While the projective 3D reconstruction algorithm is basically composed of a cascade of F-matrix estimation and triangulation blocks, a summary is deemed necessary for sake of completeness.

Algorithm: Projective 3D Reconstruction

Input: Feature correspondences.

Output: Camera matrices and sparse 3D structure as a 3D point cloud.

1. Estimate the F-matrix from feature correspondences.
2. Recover the camera matrices from Equation 3.8.
3. Estimate the structure via optimal triangulation on inliers to the F-matrix.
4. **Optional:** Refine the result by *bundle adjustment* (See Section 3.6).

3.3 Metric 3D Reconstruction from Two Views

The metric reconstruction procedure is exactly the same as projective reconstruction, except for the recovery of the camera matrices. The estimation of metric camera matrices is considerably more complex than the projective case, due to special constraints arising from the requirements of metric ambiguity level. Knowledge of F-matrix guarantees the recovery of two valid projective camera matrices. However, a metric camera matrix has to be decomposable as [10]

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}], \tag{3.26}$$

where **K** is an upper triangular matrix with positive diagonal elements *(calibration matrix*, defined in Section 3.4) and **R** is a rotation matrix. A metric camera matrix is related to F-matrix through an intermediate entity called *essential matrix* (E-matrix). The relation is [10]

$$\mathbf{E} = \mathbf{K}'^{\mathbf{T}}\mathbf{F}\mathbf{K} = [\mathbf{t}]_{\times}\mathbf{R} \tag{3.27}$$

for a camera pair **P=K[I|0]** and **P'=K'[R|t]**. In this case**, R** and **t** stand for the relative position and orientation of **P'** with respect to a camera **P** at the origin of the coordinate system.

In [30], metric camera matrix estimation problem is posed as the minimization of the cost $\|\mathbf{R}[-\mathbf{t}]_x - \mathbf{E}^T\|$ subject to the constraint that **R** is a rotation matrix. This amounts to finding the closest valid E-matrix to the E-matrix estimated via Equation 3.27. The problem is solved in 3 steps. First, an intermediate value, $\mathbf{t_s}$ is computed by solving the equation

$$\mathbf{E}^T\mathbf{t_s} = \mathbf{0}. \tag{3.28}$$

The ambiguity in the sign is resolved by multiplying the result with the sign correction term s_1 which is defined as

$$s_1 = \mathrm{sgn}\left(\sum_i \left(\mathbf{t_s} \times \mathbf{x_i'}\right)\left(\mathbf{E}\mathbf{x_i}\right) \right). \tag{3.29}$$

The next step is the solution of the minimization problem. The problems of the form $\|\mathbf{RC-D}\|$, subject to the constraint that **R** is a rotation matrix is solved by finding the eigenvector of the smallest eigenvalue of the following matrix [30]

$$\mathbf{B} = \sum_{i=1}^{3} \mathbf{B_i}\mathbf{B_i^T}, \tag{3.30}$$

where

$$\mathbf{B_i} = \begin{bmatrix} \mathbf{0} & (\mathbf{c_i} - \mathbf{d}_i)^{\mathrm{T}} \\ \mathbf{d_i} - \mathbf{c_i} & [\mathbf{d_i} + \mathbf{c_i}]_{\times} \end{bmatrix}$$

$$\mathbf{C} = [\mathbf{c_1} \quad \mathbf{c_2} \quad \mathbf{c_3}] \tag{3.31}$$

$$\mathbf{D} = [\mathbf{d_1} \quad \mathbf{d_2} \quad \mathbf{d_3}].$$

The solution, **q** is related to **R** as

$$\mathbf{R} = \begin{bmatrix} q_1^2 + q_2^2 - q_3^2 - q_4^2 & 2(q_2 q_3 - q_1 q_4) & 2(q_2 q_4 + q_1 q_3) \\ 2(q_2 q_3 + q_1 q_4) & q_1^2 - q_2^2 + q_3^2 - q_4^2 & 2(q_3 q_4 - q_1 q_2) \\ 2(q_2 q_4 - q_1 q_3) & 2(q_3 q_4 + q_1 q_2) & q_1^2 - q_2^2 - q_3^2 + q_4^2 \end{bmatrix}, \tag{3.32}$$

where q_i are the individual components of **q**.

Finally, **t** is recovered by multiplying $\mathbf{t_s}$ by another sign correction factor, s_2 which is computed by

$$s_2 = \mathrm{sgn}\left(\sum_i (\mathbf{t_s} \times \mathbf{x_i'})(\mathbf{x_i'} \times \mathbf{R}\mathbf{x_i}) \right) \tag{3.33}$$

for all ($\mathbf{x_i}$;$\mathbf{x'_i}$) that satisfy

$$\frac{\left\| \mathbf{x_i'} \times \mathbf{R}\mathbf{x_i} \right\|}{\left\| \mathbf{x_i'} \right\| \left\| \mathbf{x_i} \right\|} \geq \alpha \tag{3.34}$$

for a small α. If no such feature pair exists, it implies that $\mathbf{t=0}$.

E-matrix can also be estimated directly from point correspondences using a RANSAC-based approach with a normalized 5-point algorithm [71], similar to the one described in Section 3.2.

The algorithm for the decomposition of E-matrix into translation and rotation terms can be briefly summarized as below [30].

`Algorithm: E-Matrix Decomposition`

Input: E-matrix.

Output: Rotation and translation of the second camera of a camera pair, with respect to the first camera.

1. Compute $\mathbf{t_s}$ via Equations 3.28, and 3.29
2. Estimate **R** by finding the **q** that minimizes $\|\mathbf{Bq}\|$, where **B** is defined by Equation 3.31**.**
3. Estimate the sign correction term from Equation 3.33 to obtain **t**.

3.4 Self-Calibration

Metric reconstruction requires the knowledge of **K**, *calibration matrix* of the camera. Calibration matrix defines the mapping between the camera and image coordinate systems. For a pin-hole camera, it is of the form

$$\mathbf{K} = \begin{bmatrix} f & s & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.35}$$

Individual elements of the matrix and assumptions about their values for commonly available equipment are listed below [10]:

- **Focal length (f):** Distance of camera center to image plane.
- **Aspect ratio (α):** Aspect ratio of pixel edges in case of non-square imaging elements. It is often assumed to be unity.
- **Skew (s):** Indicates the skewing of the imaging elements in the camera. It is safe to assume it to be zero.
- **Principal point offset ($\mathbf{p_x}$, $\mathbf{p_y}$):** The translation between the origin of the camera and image coordinate systems. A common assumption is the mid-point of the image.

In the scope of this dissertation, self-calibration problem involves the recovery of calibration parameters by using only feature correspondences. To this aim, many techniques exploit the relation between the absolute dual quadric and its projection [10]. The absolute dual quadric is an entity which has the form

$$\mathbf{Q}^{*}_{\infty\mathbf{M}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.36}$$

in a metric coordinate system. In a projective system, it is deformed into

$$\mathbf{Q}^{*}_{\infty\mathbf{P}} = \mathbf{H}\mathbf{Q}^{*}_{\infty\mathbf{M}}\mathbf{H}^{\mathbf{T}}, \tag{3.37}$$

where **H** stands for the transformation between the metric and projective coordinate systems, and can be recovered through the singular value decomposition of $\mathbf{Q}^{*}_{\infty\mathbf{P}}$. **H** upgrades the reconstruction to metric ambiguity level via Equation 3.2; hence, known as *rectifying homography*. **K** can be computed from a metric camera matrix, by RQ decomposition of its first 3x3 submatrix.

$\mathbf{Q}^{*}_{\infty\mathbf{P}}$ is related to **K** as [10]

$$\mathbf{K}\mathbf{K}^{\mathbf{T}} \approx \mathbf{P}\mathbf{Q}^{*}_{\infty\mathbf{P}}\mathbf{P}^{\mathbf{T}}. \tag{3.38}$$

Zero-valued elements of $\mathbf{K}\mathbf{K}^{\mathbf{T}}$ are used to establish an equation system linear in the elements of $\mathbf{Q}^{*}_{\infty\mathbf{P}}$, as in Equation 3.9. $\mathbf{Q}^{*}_{\infty\mathbf{P}}$ is symmetric and has 10 degrees of freedom; hence, at least 10 equations are necessary for a unique solution. Besides, $\mathbf{Q}^{*}_{\infty\mathbf{P}}$ is rank-3, therefore the component corresponding to the smallest singular value should be annihilated.

In this work, two self-calibration algorithms are employed, whose details are explained in the following sections.

Practical Self-Calibration

The approach proposed in [2] utilizes the empirical observation that Equation 3.39 below is a good initial estimate for the calibration parameters:

$$\mathbf{K_{initial}} = \begin{bmatrix} w+h & 0 & \frac{w}{2} \\ 0 & w+h & \frac{h}{2} \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.39}$$

In this equation, w and h are the width and height of the image. Therefore, when the available camera matrices are left-multiplied with the inverse of the initial estimate, focal length of the normalized calibration matrix should be on the order of unity, and the principal point should be close to the origin. This information, along with the assumption of constant camera parameters, in addition to zero skew and unity aspect ratio, and a set of empirical weights, can be used to set up the following equation system [2]:

$$\begin{aligned}
&\frac{1}{9\upsilon}\left(\mathbf{p_1}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_1}^{\mathbf{T}} - \mathbf{p_3}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_3}^{\mathbf{T}}\right) = 0 \\
&\frac{1}{9\upsilon}\left(\mathbf{p_2}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_2}^{\mathbf{T}} - \mathbf{p_3}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_3}^{\mathbf{T}}\right) = 0 \\
&\frac{1}{0.2\upsilon}\left(\mathbf{p_1}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_1}^{\mathbf{T}} - \mathbf{p_2}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_2}^{\mathbf{T}}\right) = 0 \\
&\frac{1}{0.1\upsilon}\left(\mathbf{p_1}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_2}^{\mathbf{T}}\right) = 0 \\
&\frac{1}{0.1\upsilon}\left(\mathbf{p_1}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_3}^{\mathbf{T}}\right) = 0 \\
&\frac{1}{0.01\upsilon}\left(\mathbf{p_2}\mathbf{Q}^*_{\infty\mathbf{P}}\mathbf{p_3}^{\mathbf{T}}\right) = 0,
\end{aligned} \tag{3.40}$$

where $\mathbf{p_i}$ represents the i^{th} row of the camera matrix $\mathbf{P}$, and υ stands for an unknown scale factor. At least 3 frames are required to provide sufficient

constraints to solve for the unknown $\mathbf{Q}^*_{\infty P}$ uniquely, but more frames make the estimates more robust. The initial value of υ is set to 1, and the equation system is solved iteratively, by setting υ as $\mathbf{p}_3(\mathbf{Q}^*_{\infty P})\mathbf{p}_3^T$, until the value of υ converges.

Below is a summary of the technique described in this section [2].

Algorithm: Practical Self-Calibration

Input: At least 2 camera matrices not at the origin of the coordinate system.

Output: Rectifying homography **H**.

1. Normalize the camera matrices with $\mathbf{K}^{-1}_{\text{initial}}$.
2. Solve for $\mathbf{Q}^*_{\infty P}$ and υ until convergence.
3. Find the closest rank 2 approximation to $\mathbf{Q}^*_{\infty P}$.
4. Compute the rectifying homography.

Self Calibration Using F-Matrix

A radically different approach, proposed by Mendonça et al. [27] exploits the fact that two singular values of an E-matrix are equal, to solve directly for the calibration matrix. In order to achieve this, the algorithm aims to find a **K**, which minimizes the difference of the eigenvalues of the E-matrix **E**, computed from Equation 3.27, by minimizing the cost function

$$C(\mathbf{K}) = \sum_{ij} w_{ij} \frac{\sigma_{ij}^1 - \sigma_{ij}^2}{\sigma_{ij}^2} \tag{3.41}$$

for constant **K** via a non-linear minimization algorithm, such as *Levenberg-Marquardt*. w_{ij} is a weight signifying the reliability of the F-matrix between the i^{th}

and j^{th} images, $\mathbf{F_{ij}}$, and σ^{1}_{ij} and σ^{2}_{ij} are the first and second singular values of $\mathbf{KF_{ij}K^T}$. Equation 3.39 is used as the initial estimate. Although the algorithm is capable of handling more frame pairs, assuming that the only unknown parameter is focal length, a single frame pair is sufficient to find a solution.

3.5 Trajectory Segmentation

In a scene with dynamic elements, i.e., independently moving bodies, in addition to static background, each motion defines an F-matrix, which imposes an epipolar constraint on all feature pairs conforming to that motion. Conventional F-matrix estimation algorithms are capable of recovering only one of these F-matrices, usually the one belonging to the element which has the largest number of features. If individual reconstructions of the background and the dynamic elements are required, this is an undesirable situation. Geometric segmentation approach allows both the labeling of the feature pairs and their trajectories, and the recovery of the F-matrices corresponding to the motions present in the scene.

The geometric segmentation approach proposed in [72] performs multiple passes over a feature correspondence set, extracting one of the F-matrices at each pass. Once an F-matrix is recovered, inliers to this model are removed from the correspondence set, and the next pass is performed by using only the remaining outliers. In order to achieve spatial coherency, the inliers are subjected to a threshold with respect to their Euclidean distance to the centroid of the inliers, i.e.,

$$\left\|\mathbf{x_i}-\mathbf{c}\right\|^2+\left\|\mathbf{x_i'}-\mathbf{c'}\right\|^2<w\left(\sigma^2+\sigma'^2\right), \tag{3.42}$$

where ($\mathbf{c};\mathbf{c}'$) are the centroid of the inliers to the current F-matrix in each image, w a scale factor and (σ^2, σ'^2) are the variances of the inliers. If a feature pair does

not satisfy this condition, it is returned to the outlier set. The procedure is terminated when all trajectories are labeled, or the remaining correspondences do not provide any more reliable F-matrices.

An important issue in practice is the automatic selection of the frame pairs, or *key frames*, which facilitate the estimation of a reliable F-matrix that will lead to an accurate segmentation. This is achieved by selecting a frame as the first key frame, and seeking a suitable second frame [75]. For each candidate, two competing models, an F-matrix, and a homography are evaluated. Homography prevails when all features belong to a single plane, or the camera motion between the images does not involve a displacement of the camera center. Any of these cases, or their close approximations, such as small translations, are not suitable to F-matrix estimation, and consequently geometric segmentation via epipolar criterion. Both models are evaluated by GRIC, defined in Equation 3.12.

The geometric segmentation algorithm is presented below [72].

`Algorithm: Geometric Segmentation`

Input: Trajectories.

Output: Labeled trajectories and corresponding F-matrices, key frame pairs.

1. Mark the first frame as key frame.
2. Until all frames are processed
    - ***a.*** Compute an F-matrix by using the feature correspondences between the last key frame and the current frame.
    - ***b.*** Compute a homography by using the feature correspondences between the last key frame and the current frame.
    - ***c.*** Compare two models by GRIC. If F-matrix prevails, mark the current frame as a key frame.

d. Proceed to next frame.

3. For each successive key-frame pair, until all trajectories are classified or no more reliable F-matrices can be estimated

a. Estimate F-matrix via the feature pairs from unlabeled trajectories.

b. Check spatial coherency of the inliers.

c. Label the trajectories corresponding to the inliers

3.6 Multi-View 3D Reconstruction

When more than two views are available, the basic two-view reconstruction algorithm should be upgraded to accommodate for the additional data. In this thesis, mainly the sequential reconstruction approach described in [2] is adopted. This approach uses the existing structure estimate to locate the cameras belonging to new frames, and gradually builds a 3D point cloud by accumulating the structure presented by each consecutive frame. Finally, the recovered camera matrix and structure estimates are refined via a non-linear optimization stage, known as *bundle adjustment*.

3.6.1 Sequential Reconstruction

The sequential reconstruction algorithm [2] starts with an initial structure and camera matrix estimate, obtained from the first two frames of the sequence. These frames are denoted as *Frame*-1 and *Frame*-2 in the example in Figure 3.3. If some of this initial structure is also visible in the new frame, *Frame*-3 in the example, it is possible to establish correspondences between the visible part, and the 2D features in the image by using the trajectories with points in all 3 frames. If a sufficient number of 2D-3D correspondences are available, an estimate of the camera matrix for *Frame*-3 can be obtained, with a method akin to robust

homography estimation, discussed in Section 3.2.1. The basic equations are derived from the relation **x≈PX** as

$$\begin{aligned} &-Xp_{21}-Yp_{22}-Zp_{23}-Wp_{24}+xXp_{31}+xYp_{32}+xZp_{33}+xWp_{34}=0 \\ &-Xp_{11}-Yp_{12}-Zp_{13}-Wp_{14}+yXp_{31}+yYp_{32}+yZp_{33}+yWp_{34}=0, \end{aligned} \tag{3.43}$$

where $\mathbf{X}=[X\ Y\ Z\ W]^T$ and p_{ij} are the elements of **P**. 6 correspondences (11 constraints) are required to generate a hypothesis. Inliers are determined by *reprojection error*, which is defined as [10]

$$e_{reprojection} = \left\| \mathbf{x_i} - \frac{\mathbf{x_{ip}}}{x_{ip3}} \right\|^2, \tag{3.44}$$

where

$$\mathbf{x_{ip}} \approx \mathbf{P}\mathbf{X_i} \tag{3.45}$$

and x_{ip3} is its 3rd component, for a 2D-3D correspondence ($\mathbf{x_i}$; $\mathbf{X_i}$).

Once the camera matrix belonging to the new frame is estimated, the feature pairs between *Frame*-2 and *Frame*-3 are triangulated to yield a 3D point cloud. This point cloud is composed of two sets of points. One set belongs to the portion of the structure not visible in *Frame*-1, but available in *Frame*-2 and 3. This set is added to the current structure estimate. The other set is composed of the points that are visible in both *Frame*-1 and 2. They are used to refine the existing structure estimate by weighted-averaging, in which the weights being the inverse

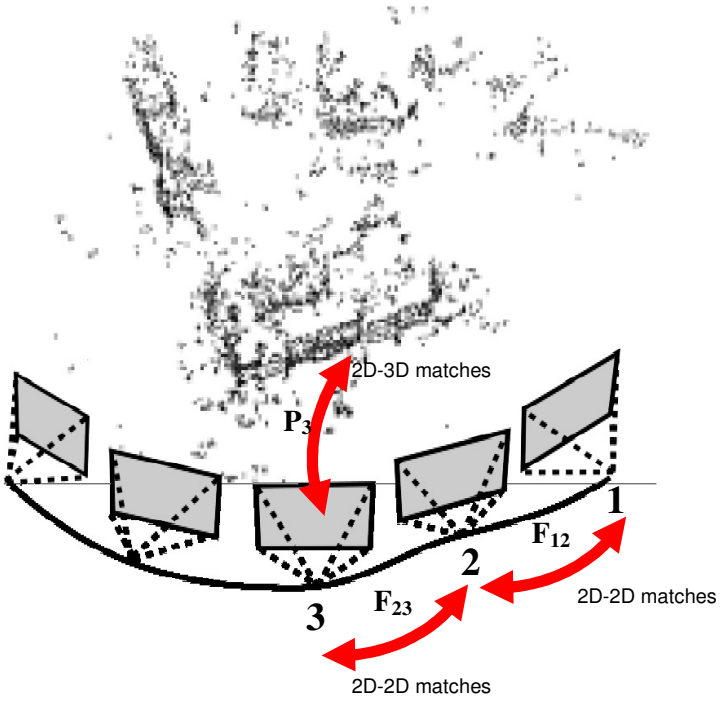


Figure 3.3: Sequential multi-view reconstruction.

of the reprojection error over the trajectories of the features in the processed frames.

Algorithm: Sequential 3D Reconstruction

Input: Feature trajectories.

Output: Sparse structure and camera matrix estimates.

1. Compute initial reconstruction and camera matrices.
2. For all frames
    - ***a.*** Estimate the camera matrix by using 2D-3D correspondences.
    - ***b.*** Compute the 3D points corresponding to the feature pairs from the current and the previous frame.
    - ***c.*** Update the structure.
    - ***d.*** Proceed to next frame.

3.6.2 Bundle Adjustment

The sequential reconstruction algorithm is an alternating scheme that estimates the camera matrix from the structure, and then, uses this camera matrix (or equivalently, F-matrix) to update the structure for all consecutive frame pairs. In contrast, bundle adjustment is the joint optimization of structure and camera parameters for all frames simultaneously, with respect to reprojection error, defined in Equation 3.44. The minimization problem can be expressed as

$$\text{minimize} \sum_i \sum_j d_{reprojection}\left(\mathbf{x_{ij}}, \mathbf{X_i}, \mathbf{P_j}\right) \quad \text{wrt } \{\mathbf{X_i}\}_i, \{\mathbf{P_j}\}_j, \tag{3.46}$$

where $\mathbf{X_i}$ denotes the i^{th} structure point, $\mathbf{P_j}$, the j^{th} camera matrix, $\mathbf{x_{ij}}$, the projection of i^{th} point to j^{th} view, and $\{\mathbf{X_i}\}_i$ and $\{\mathbf{P_j}\}$j, the sets of all structures and camera matrices, respectively.

The minimization is typically performed by *Levenberg-Marquardt* technique [2], a variant of Newton iteration. $\mathbf{\Delta}$, the update term to the solution at each step, is computed by the equation

$$\mathbf{N\Delta = Je}, \tag{3.47}$$

where **J** is the Jacobian of the reprojection error with respect to structure and camera parameters, **e**, the difference between the tracked and reprojected features, and **N** is defined as

$$\mathbf{N = J^T J} + \lambda\mathbf{I}, \tag{3.48}$$

where **I** denotes the identity matrix. The parameter λ determines the amount of divergence from Newton's method. A low value for λ means a Newton-like operation, and a high λ makes it closer to steepest-descent (see Section 4.4). If an update reduces the error, it is accepted, and λ is reduced. Otherwise, it is rejected, and λ is increased. Each step is guaranteed to provide an improvement, as eventually λ increases to the point that the operation approximates steepest-descent.

The computational cost of the minimization operation can be significantly reduced by recognizing the fact that the reprojection error for $\mathbf{x_{ij}}$ depends only on the i^{th} structure point and the j^{th} camera matrix [46]. Therefore, the Jacobian has zero

values for all other structure points and camera parameters, hence is a very sparse matrix. The structure of the matrix is depicted in Figure 3.4 [2]. The non-zero blocks can be expressed as follows [46]:

$$\begin{aligned}\mathbf{U_j} &= \sum_i \left(\frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{P_j}}\right)^T \frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{P_j}} \\ \mathbf{V_i} &= \sum_j \left(\frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{X_i}}\right)^T \frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{X_i}} \\ \mathbf{W_{ji}} &= \left(\frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{P_j}}\right)^T \frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{X_i}}.\end{aligned} \tag{3.49}$$

Similarly, the right side of the equation can be partitioned as [46]

$$\begin{aligned}\varepsilon(\mathbf{P_j}) &= \sum_i \left(\frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{P_j}}\right)^T \varepsilon_{ji} \\ \varepsilon(\mathbf{X_i}) &= \sum_j \left(\frac{\partial \mathbf{x_{ij}}}{\partial \mathbf{X_i}}\right)^T \varepsilon_{ji},\end{aligned} \tag{3.50}$$

where

$$\boldsymbol{\varepsilon} = \mathbf{Je}. \tag{3.51}$$

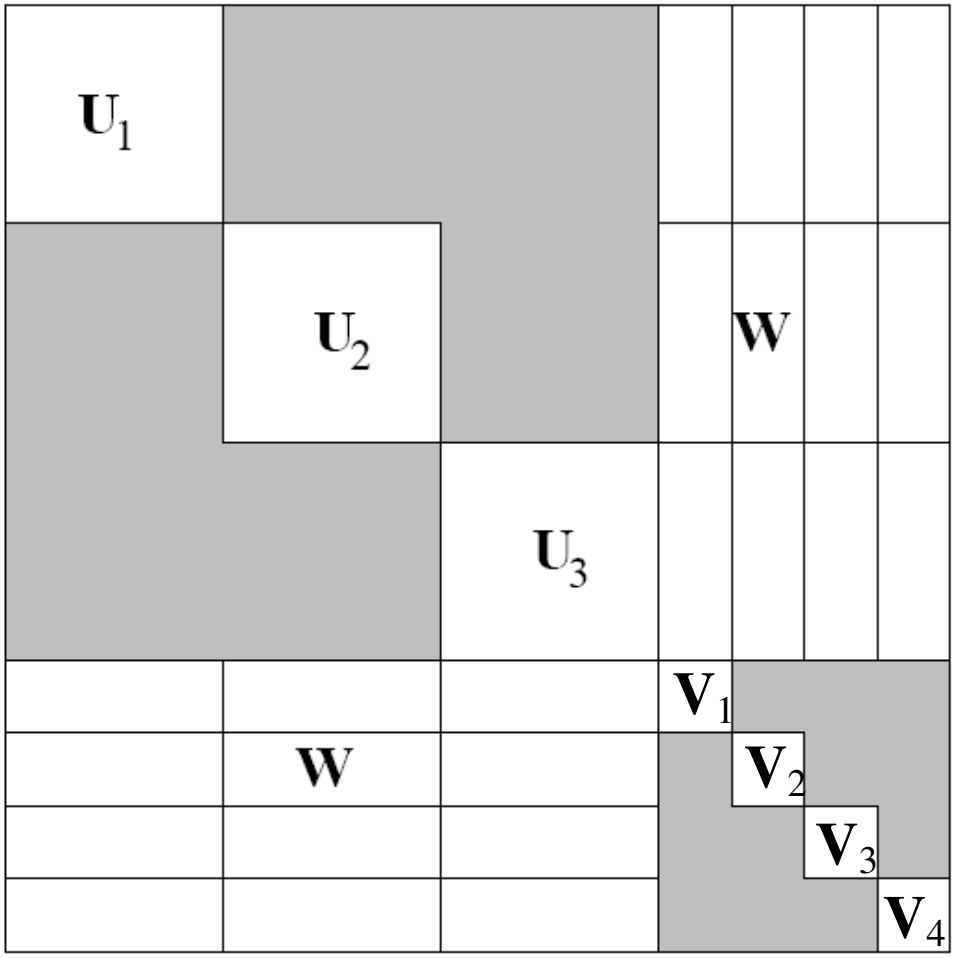


Figure 3.4: Structure of the Jacobian. Shaded regions represent zero-valued entries.

Then, the solution of Equation 3.47 can be obtained by solving [46]:

$$\begin{aligned}\left(\mathbf{U}-\mathbf{W}\mathbf{V}^{-1}\mathbf{W}^{\mathbf{T}}\right)\boldsymbol{\Delta}_{\mathbf{P}} &= \boldsymbol{\varepsilon}(\mathbf{P})-\mathbf{W}\mathbf{V}^{-1}\boldsymbol{\varepsilon}(\mathbf{X}) \\ \boldsymbol{\Delta}_{\mathbf{x}} &= \mathbf{V}^{-1}\left(\boldsymbol{\varepsilon}(\mathbf{X})-\mathbf{W}^{\mathbf{T}}\boldsymbol{\Delta}_{\mathbf{P}}\right)\end{aligned} \tag{3.52}$$

where $\boldsymbol{\Delta}_{\mathbf{P}}$ and $\boldsymbol{\Delta}_{\mathbf{X}}$ are update terms for the camera and structure parameters. The sparse bundle adjustment algorithm is summarized below [46].

`Algorithm: Sparse Bundle Adjustment`

Input: Trajectories, structure and camera matrix estimates.

Output: Refined structure and camera matrix estimates.

1. Until the error converges
    - ***a.*** Compute the intermediate matrices **U, W, V, ε(P)** and **ε(X)** via Equation 3.49 and 3.50.
    - ***b.*** Add $(1+\lambda)$ to the diagonal entries of **U** and **V**.
    - ***c.*** Compute $\mathbf{V}^{-1}$.
    - ***d.*** Solve Equation 3.52 for $\boldsymbol{\Delta}_{\mathbf{P}}$ and $\boldsymbol{\Delta}_{\mathbf{X}}$.
    - ***e.*** Update the parameters, and compute the new error.
    - ***f.*** If error decreases, accept the update and decrease the value of λ. Go to Step 1.a.
    - ***g.*** If error increases, reject the update and increase the value of λ. Go to Step 1.b.

3.7 Prioritization

Any *N*-frame image (wide-baseline) or video (narrow-baseline) sequence has $\frac{N(N+1)}{2}$ frame pairs from which information on structure and camera matrices could be gathered. Out of these, the sequential reconstruction algorithm of Section 3.6 uses only a subset of *N* pairs, and the sole criterion in the selection of this subset is the consecutiveness of the frames. This raises an interesting question about whether it is possible to achieve better results with a subset selected by some criteria more relevant to the reconstruction quality. The answer should favor a subset as small as possible to minimize the computational load, while still allowing the recovery of as much of the structure as feasible.

For the video case, the question goes beyond being a mere thought exercise, as unlike image sequences, the default temporal order of a video sequence certainly does not provide the necessary baseline length for reliable structure and camera matrix estimates. Frame skipping can be employed as an ad-hoc solution, but obviously, this practice does not guarantee appropriate frame pairs unless there is some prior information about camera motion justifying this assumption. Moreover, blindly discarding frames excludes potentially good frame pairs from the reconstruction process, thus simply wastes information.

Since the reconstruction is performed sequentially, an equally important issue is the order in which the frame pairs are processed. In order to ensure convergence to a satisfactory solution, a reliable intermediate structure estimate should be achieved early on, as otherwise, for a new frame poor structure estimate would lead to incorrect camera localization, and in turn, a poor structure update, creating a vicious cycle. Therefore, the problem is not only selecting a good set of frame pairs, but also establishing a good processing order.

The discussion above implies that the following criteria should be taken into account, while designing a frame pair selection procedure.

- **Fast convergence to a reliable estimate:** Since the quality of the subsequent reconstructions depends on the current reconstruction, the frame pairs that are likely produce a reliable structure estimate should be processed first.
- **Fast recovery of the entire structure:** The number of reconstructed 3D points should be maximized, while processing minimum number of frame pairs, to obtain a good sparse description of the scene in a computationally efficient manner.
- **Assessment of all possible pairs:** The number of frame pairs assessed by the procedure should be as high as possible, ideally the entire set of pairs, to improve the probability of discovering the best pair set and processing order.

In the literature, previously studied frame pair evaluation metrics are GRIC, and the lower bound on the structure estimation error [73]. Since the former is designed only to avoid degenerate cases in F-matrix estimation, it satisfies the above criteria only to a limited extent. Moreover, it is computationally unfeasible to estimate the F-matrix for every frame pair. The latter considers both the number of reconstructed points and their reliability, in the form of the estimate covariance matrix. While this metric seems appealing, since it covers the first two criteria stated above in a theoretically sound manner, it is designed to assess the suitability of a frame pair as the initial estimate to a bundle adjustment procedure, and requires the computation of 3D structure. This generates a computational load that forbids its use in a scheme which can assess all frame pairs.

In this work, a novel metric is proposed, in terms of the weighted sum of the baseline distance and a nonlinear function of the number of matches. This metric, in a way, approximates the one described in [73], as the baseline distance is related to the covariance matrix of the reconstruction. However, it only requires relative poses of the cameras and the number of feature matches between the frame pairs. The pose estimates for the entire sequence are computed by using a reference reconstruction obtained from two frames, and 3D-2D correspondences, via the method described in Section 3.6.1. The number of matches between the frames is obtained from the trajectories.

It should be noted that the use of baseline distance implies a metric reference reconstruction. However, even a rough estimate of the calibration matrix is sufficient, as the final processing order is essentially robust to reasonable calibration errors. This estimate is obtained from the pair used in the reference reconstruction, by Mendonça's method, which is explained in Section 3.4. However, it is possible to use the key-frames determined by the segmentation module, when trajectory segmentation is necessary, or even the estimate given in Equation 3.39.

The priority metric, p, utilized in the algorithm to evaluate the feasibility of a frame pair for reconstruction is defined as

$$p = d + \frac{\alpha}{1 + \exp\left(\beta\left(n - \varphi\right)\right)}, \tag{3.53}$$

where d is the baseline distance between the cameras, n the number of feature matches, and α, β and φ are the design parameters of the sigmoid function appearing in the second term. The rationale behind Equation 3.53 is the fact that the non-linear (sigmoidal) weighting keeps the contribution of the second term

within a bound, when there is a relatively small or large number of matching features. It should be noticed that the existence of some other metrics delivering a comparable performance is possible. However, in the experiments, the proposed metric is demonstrated to achieve a performance superior to its most obvious alternatives.

Below is a recapitulation of the proposed frame pair prioritization algorithm.

`Algorithm: Frame Pair Prioritization`

Input: Feature trajectories, and a key-frame pair.

Output: Ordered frame pairs.

1. Compute the calibration matrix via Mendonça's method.
2. Compute the reference structure by using the metric reconstruction algorithm described in Section 3.3.
3. Estimate the poses of all frames with respect to the reference structure via the camera matrix estimation algorithm from 2D-3D correspondences, described in Section 3.6.1.
4. For all frame pairs, evaluate the prioritization metric given in Equation 3.53.
5. Sort the frame pairs with respect to the prioritization metric.

3.8 Prioritized Sequential 3D Reconstruction

As mentioned in Section 3.1, the previous sections of Chapter 3 are aimed at introducing the building blocks of a novel, estimate-fusion type MFSfM technique, *prioritized sequential 3D reconstruction*. The algorithm is actually a generalization of the conventional sequential reconstruction method described in

Section 3.6, to the case in which the consecutive ordered frame pairs might not share a common frame. It maintains multiple sequential reconstructions, built from mutually exclusive subsequences of the video sequence. The processing order determines which reconstructions grow, which reconstructions are merged, and when the merger takes place.

The following definitions should enhance the clarity of the presentation of the algorithm.

Definition 3.1: A *sub-estimate* is a structure estimate obtained by the triangulation of the matching features in a single frame pair.

Definition 3.2: A *sub-reconstruction* is an intermediate structure estimate obtained from a collection of *sub-estimates* belonging to a subset of frames of the video sequence. Two distinct sub-reconstructions cannot have any common frames. The global motion and structure estimate is computed by merging the sub-reconstructions.

The algorithm proceeds through the ordered frame pairs, taking one of the three possible actions at each frame pair.

Initiate: If both frames are encountered for the first time, i.e., none of them was used previously in any of the existing sub-reconstructions, a new sub-reconstruction is initiated for the frame pair by using the two-view reconstruction algorithm of Section 3.2.

Add: If one of the frames is already used in a sub-reconstruction, the corresponding sub-estimate is added to that sub-reconstruction, via the sequential

reconstruction algorithm described in Section 3.6.1. This action can also be invoked by a frame pair, in which both frames belong to the same sub-reconstruction, but added through different frame pairs. If the new sub-estimate causes a significant change in the structure, either by bringing in a large number of new 3D points, or by modifying the position of the existing points considerably, camera matrices are re-estimated from the 2D-3D correspondences. Then, the outliers in the structure are eliminated via reprojection error. This practice approximates a global minimization stage, ensuring that the quality of structure and camera matrix estimates is uniform for both previously and recently estimated entities, and computationally much more inexpensive than bundle adjustment.

Merge: One final possibility is the case when the frames in the pair are used in different sub-reconstructions. This case signals the merger of two sub-reconstructions. Since each sub-reconstruction has its own coordinate system, the merger operation requires the recovery of the 4x4 homography relating these coordinate systems. This can be achieved through a 3D extension of the 2D-2D homography estimation method described in Section 3.2.

The output of the algorithm is the projective structure and camera matrix estimates. These estimates can be upgraded to metric level by using *practical self calibration* algorithm [2] described in Section 3.4, if a metric output is required. However, for dense scene geometry representation, projective estimates are sufficient.

One remaining issue with the above algorithm is the choice of initial frame pair for the computation of the prioritization metric and ordering. This choice is constrained by the following criteria:

- Since 2D-3D correspondences are used to locate other cameras, the frame pair should allow the computation of a reliable structure.
- The estimated structure should have as many correspondences as possible with the rest of the frames in the sequence, so that the pose estimate is sufficiently good for the computation of the priority of the related pairs.

In case of the dynamic scenes, the segmentation module supplies non-degenerate frame pairs. Otherwise, a non-degenerate frame pair can be chosen randomly, among the frames which have more than a certain amount of common trajectories with the rest of the sequence.

In a long sequence, with many covered and uncovered regions, it might not be possible to find a non-degenerate frame pair that has many matches with all of the frames in the sequence. In that case, it is better to partition the sequence into subsequences, and run separate reconstruction processes with different initial frame pairs. Then, the reconstructions are merged by using the 3D-3D matches. This case can be automatically identified and resolved using the estimated metric camera trajectory for prioritization, the quality of estimates, and the number of common trajectories.

Below is a summary of the proposed prioritized sequential 3D reconstruction algorithm. The flowchart of the algorithm is depicted in Figure 3.5. The operation is illustrated with an example in Figure 3.6.

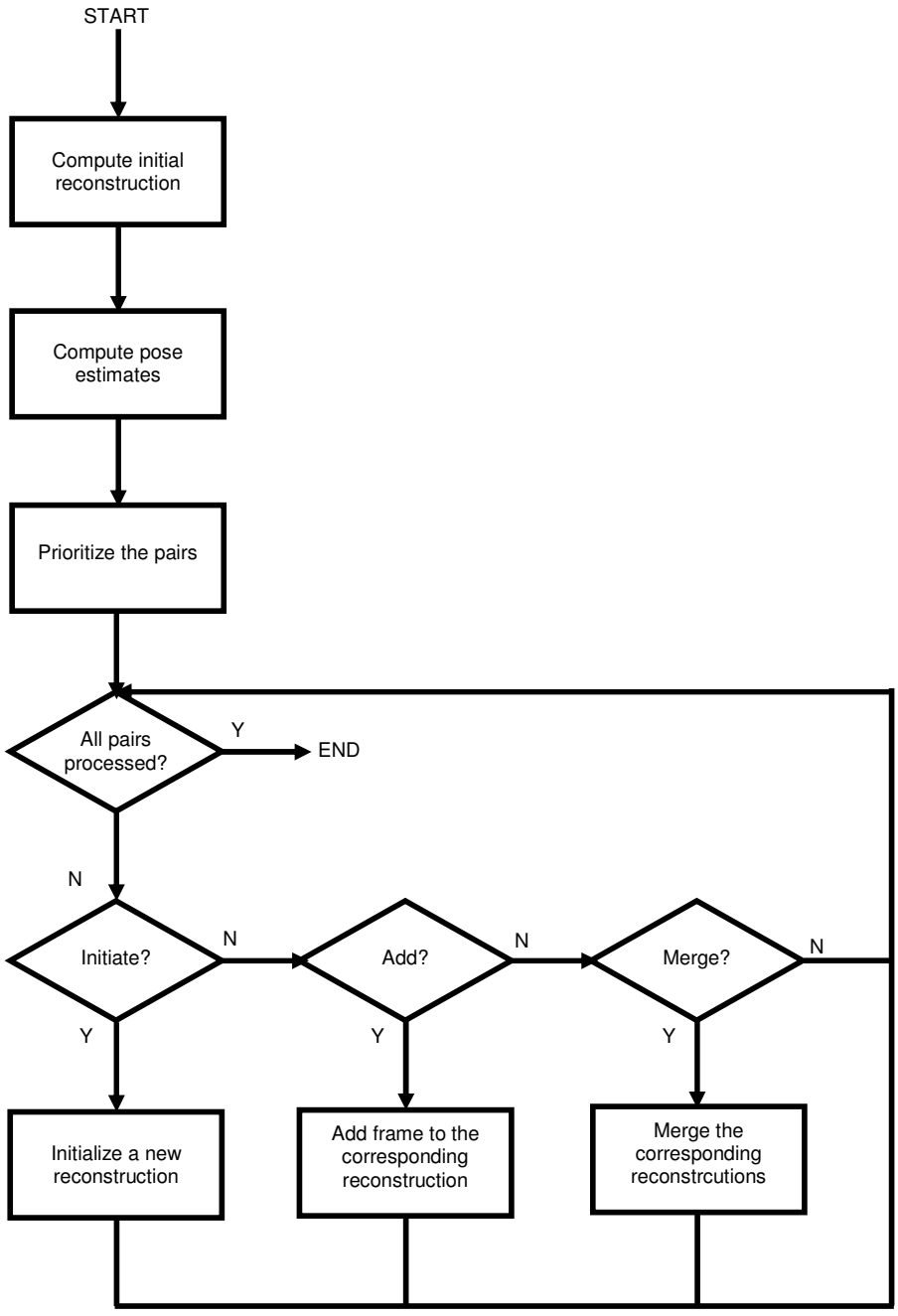


Figure 3.5: Flowchart of the prioritized sequential 3D reconstruction algorithm.

Figure 3.6: A sample reconstruction process. Frame pairs are supplied to the algorithm as an ordered pair list.

Algorithm: Prioritized Sequential 3D Reconstruction

Input: Feature trajectories.

Output: Sparse structure estimate in the form of a 3D point cloud.

1. Determine the initial frame pair.
2. Order the frame pairs.
3. While the priority metric for the current frame pair is above the threshold
    - ***a.*** If no member of the pair belongs to any of the existing sub-reconstructions, initialize a new sub-reconstruction
    - ***b.*** If one or two members of the pair belong to an existing sub-reconstruction, add the pair to this sub-reconstruction. Re-estimate the structure and camera matrices, if necessary.
    - ***c.*** If two members of the pair belong to different sub-reconstructions, merge the sub-reconstructions. Re-estimate the structure and camera matrices, if necessary.
4. Merge all remaining sub-reconstructions.
5. **Optional:** Minimize reprojection error by using bundle adjustment.

3.9 Experimental Results

In order to assess the performance of the proposed algorithm, 3 sets of experiments were conducted. In the first experiment, the reconstruction results for different prioritization metrics were compared. The second set was aimed to justify the use of multiple initial pairs for longer sequences. Finally, in the third experiment, the sparse reconstruction performance of the algorithm was compared to that of the one proposed in [2], with extension to video case, as in [5]. The

experiments were conducted on *TUB-Room*, a 240-frame synthetic sequence, *Cliff* a 109-frame sequence, and *Palace*, a 200-frame sequence. The latter two sequences were captured from typical TV broadcast with camera motion. The reconstruction quality was measured with reprojection error (Equation 3.44) of the projective reconstruction.

Figures 3.7, 3.8 and 3.9 depict sparse metric 3D reconstructions for 3 data sets, *TUB-Room, Palace* and *Cliff*, respectively.

3.9.1 Different Prioritization Metrics

In this experiment, 3 alternatives for frame pair evaluation were explored. *Nonlinear* is the metric proposed in Equation 3.53, whereas *Baseline* considers only the baseline length between the cameras. *Linear* uses the weighted average of baseline length and the number of corresponding features. The results of the experiment are presented in Table 3.1, 3.2 and 3.3.

The results identify *nonlinear* metric as a good compromise between *baseline*, which yields the most accurate reconstruction, and *linear,* which recovers the largest number of points. These observations are intuitively quite acceptable, considering the constituent terms of the metrics. *Baseline* approach does not take the number of matches into account, and uses only the pairs with the largest baseline distance. A large baseline length ensures a reliable reconstruction. However, such an approach decreases the number of matches, and the amount of structure added to the reconstruction. On the other hand, *linear* emphasizes the number of matches in the pair, and tends to make use of the pairs with a large number of matching features. While the baseline length term usually eliminates the unreliable pairs, a sufficiently high number of matches might still allow an

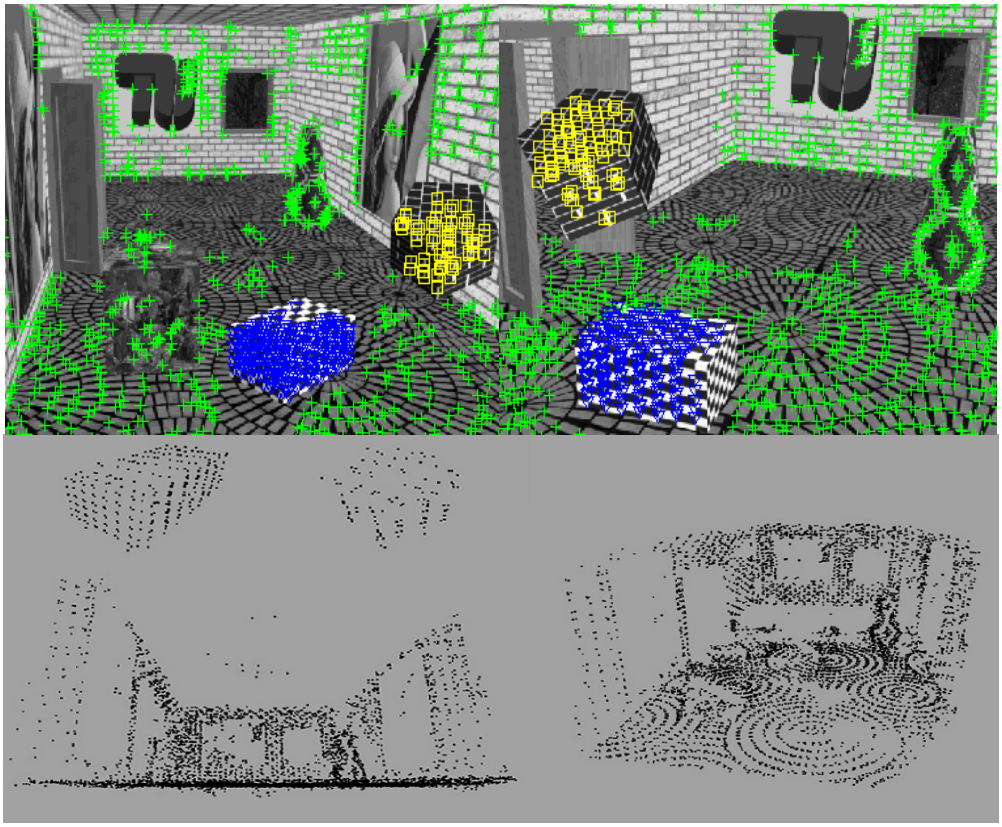


Figure 3.7: Trajectory segmentation and sparse 3D reconstruction of *TUB-Room*. *Top row:* First and last frames of the sequence, with trajectory belonging to background and 2 IMOs marked in different colors. *Middle row:* The reconstruction of IMOs. *Bottom row:* Two views of the background.

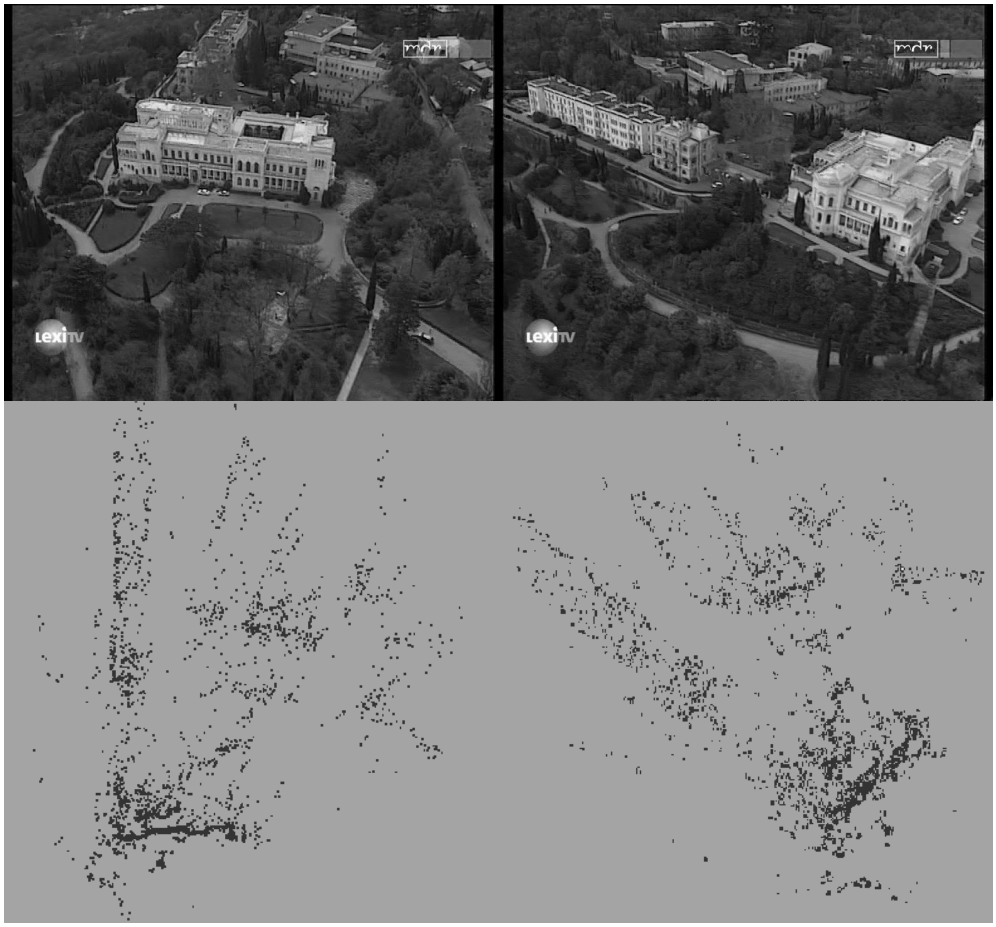


Figure 3.8: Sparse 3D reconstruction of *Palace*. *Top row:* First and last frames of the sequence. *Bottom row:* Top and top-left views.

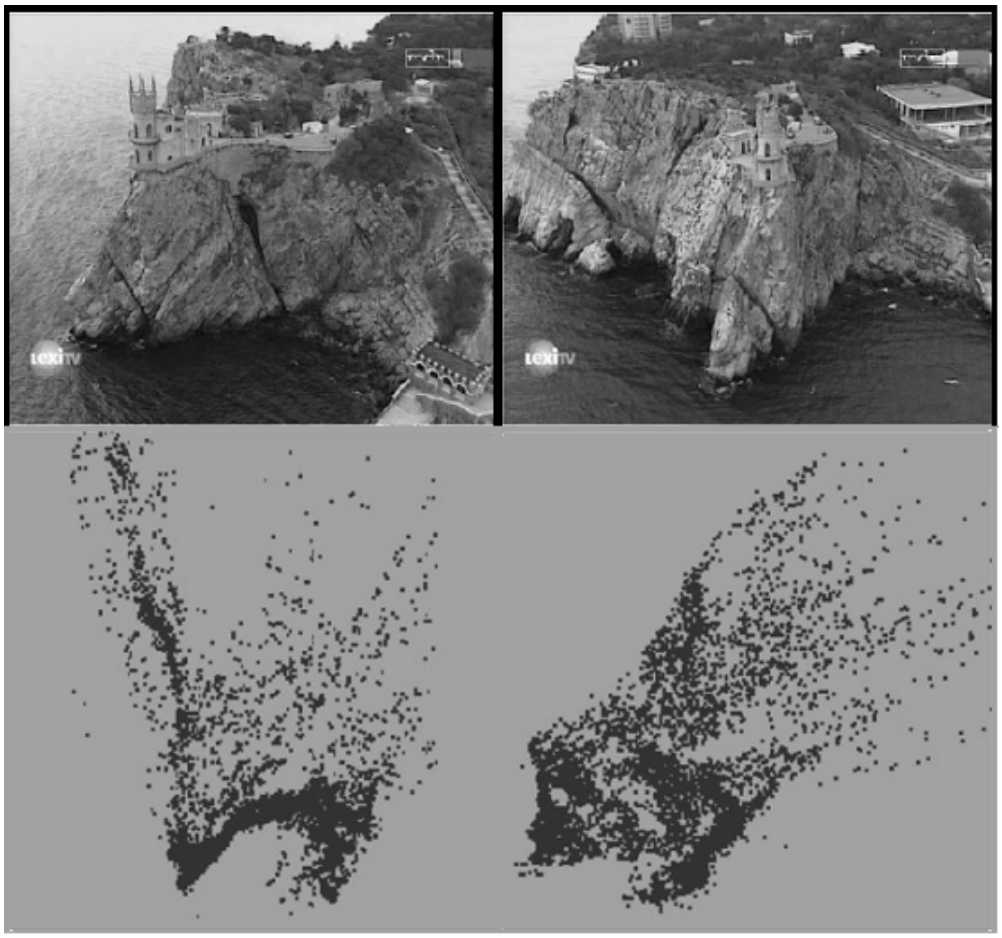


Figure 3.9: Sparse 3D reconstruction of *Cliff*. *Top row:* First and last frames of the sequence. *Bottom row:* Top and top-right views.

Table 3.1: The performance of priority metrics for *TUB-Room*

	#Frame Pairs (out of 28060)	**#Structure Points** (out of 6095)	**Average Reprojection Error** (pixels) (Equation 3.44)
Baseline	14	630	0.19
Linear	18	4961	0.58
Nonlinear	17	4716	0.27

Table 3.2: The performance of priority metrics for *Palace*

	#Frame Pairs (out of 19900)	**#Structure Points** (out of 3546)	**Average Reprojection Error** (pixels) (Equation 3.44)
Baseline	30	1284	0.73
Linear	22	3481	1.31
Nonlinear	25	2771	0.90

Table 3.3: The performance of priority metrics for *Cliff*

	#Frame Pairs (out of 5886)	**#Structure Points** (out of 8114)	**Average Reprojection Error** (pixels) (Equation 3.44)
Baseline	24	2891	0.97
Linear	17	6149	1.64
Nonlinear	20	5055	1.12

Table 3.4: Single vs. two initial reconstructions

#Reconstructions	#Frame Pairs (out of 19900)	#Structure Points (out of 3546)	Average Reprojection Error (pixels) (Equation 3.44)
1	29	2476	1.81
2	25	2771	0.90

unreliable pair into the reconstruction. The occurrence of such cases degrades the reconstruction quality. This problem is non-existent for *nonlinear*, as the cut-off effect of the sigmoid function causes the frame pairs with high number of matches effectively ordered with respect to their baseline distances. This mechanism allows the system to process more reliable frames first, and refuse the unreliable pairs, relying on the existing 3D structure estimate.

3.9.2 Multiple Initial Frames

In this experiment, *Palace*, a sequence in which the camera moves around the two faces of the building, was used to test the effect of using multiple initial reconstructions to combat against occlusions and disocclusions. The sequence was processed as two 100-frame sequences, and *nonlinear* metric was used for ordering the frame pairs. The result, presented in Table 3.4, indicates that the use of multiple reconstructions significantly improves the quality. The reason behind the success of this 2-initial reconstruction scheme is the reliability of the baseline length estimates. For a single reference camera, the camera positions in the two extremes of the sequence are estimated less accurately due to low number of correspondences, and the degraded estimates, in turn, reduce the accuracy of the baseline length estimates and the priority metric. Such a problem is avoided when multiple reference reconstructions are used.

Table 3.5: Conventional sequential reconstruction vs. proposed method

Average Reprojection Error (pixels) (Equation 3.44)	TUB-Room	Palace	Cliff
Proposed method	0.27	0.9	1.12
Conventional	2.19	3.61	2.76

3.9.3 Prioritized vs. Conventional Sequential Reconstruction

In a final experiment, to assess the competitiveness of the proposed algorithm, the structure estimates were compared with the ones obtained by [2], which is a well-known system that survived practically unchanged within the more recent algorithms, such as [74]; thus, also defines the state-of-the art. The algorithm employs the sequential reconstruction algorithm of Section 3.6, with the addition of GRIC-based key-frame selection procedure to determine the subset and order of frame-pairs to be processed [75]. For a fair comparison, the same self-calibration matrix was utilized in both algorithms, and the inlier thresholds are adjusted to recover approximately the same number of structure points. Figures 3.10, 3.11 and 3.12 depict the structure estimates. Apparently, the proposed algorithm achieves a better reconstruction, resulting with perpendicular walls in *Palace* and *TUB-Room*. In *Cliff*, there exists no discernible quality difference in the figure. However- the reprojection errors in Table 3.5 clearly indicate that the proposed algorithm outperforms the conventional method.

The superiority of the proposed method can be attributed to a multitude of reasons. First, the proposed ordering scheme is capable of assessing all available frame pairs in the sequence, thus finding a better subset and ordering. On the other

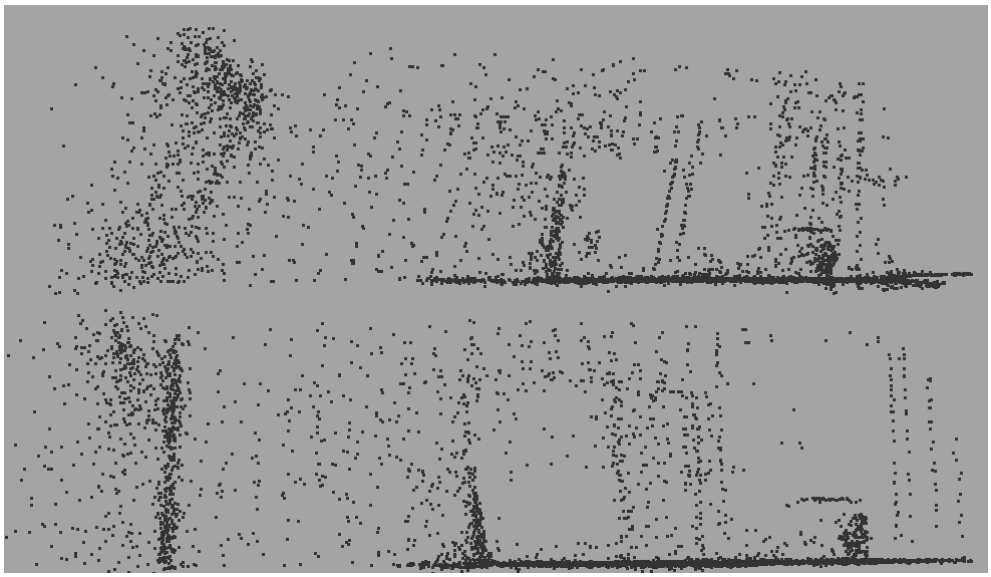


Figure 3.10: Comparison of the proposed method with [74] in *TUB-Room*. *Left:* Top view of the reconstruction by [74] .*Right:* By the proposed method.

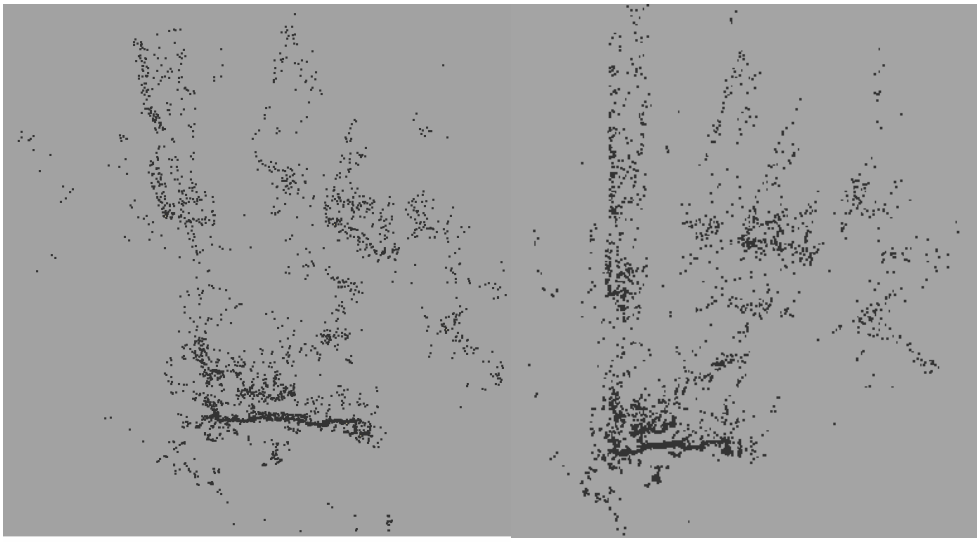


Figure 3.11: Comparison of the proposed method with [74] in *Palace*. *Left:* Top view of the reconstruction by [74] .*Right:* By the proposed method.

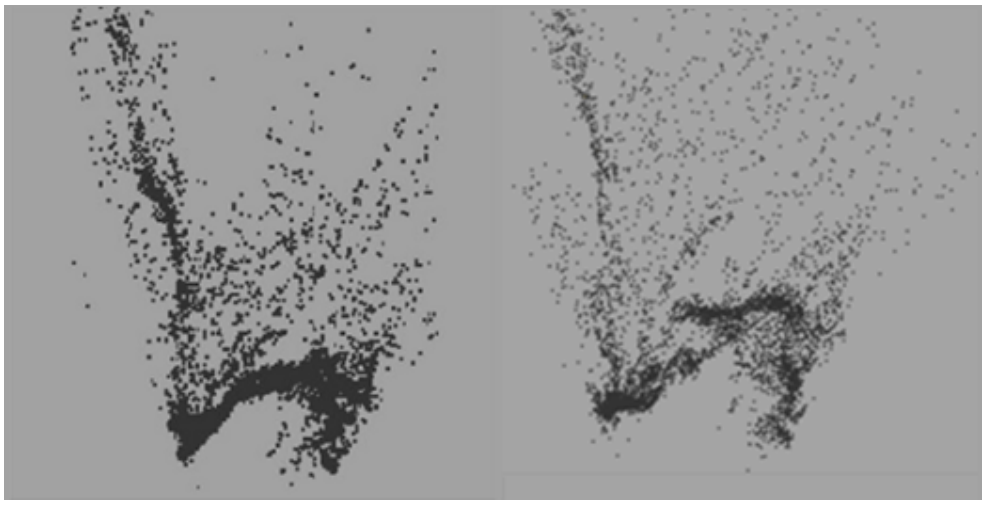


Figure 3.12: Comparison of the proposed method with [74] in *Cliff*. *Left:* Top view of the reconstruction by [74] .*Right:* By the proposed method.

hand, the GRIC-based algorithm of [75] can only evaluate N frame pairs, out of $\frac{N(N+1)}{2}$, for an N-frame sequence.

Another reason is the reliability of the selected frame pairs. The reliability is related to the baseline length, and the number of correspondences. The key-frame determination criterion of [75] usually chooses the pair with the shortest possible baseline length for given a key-frame, among all pairs for which the fundamental matrix model prevails over homography to explain the correspondences. While such an approach also results in a high number of correspondences, still, the robustness of the reconstruction is compromised, as seen in Section 3.9.1. However, in the proposed scheme, the effects of the baseline length and number of correspondences are explicitly weighted, resulting in selection of frame pairs that are more amenable to 3D reconstruction.

Finally, the reconstruction engine in the proposed algorithm provides an edge over [2], as it is capable of maintaining multiple reconstructions simultaneously. Since

each reconstruction starts from a different point in the solution space and traces a different path to the solution, the algorithm is less susceptible to a poor initial structure estimate and less prone to being trapped in local minima, when compared to GIRC-supported sequential reconstruction.

The discussion above indicates the performance advantages of the proposed scheme over the conventional method. Another issue that should be discussed is the computational cost of these gains. The performance improvement achieved by using multiple reconstructions is accompanied by a slight increase in computational complexity, due to the merger operation. As for frame pair ordering, there is actually a decrease in computational burden. This stems from the need to estimate both an F-matrix and a homography for key-frame determination by GRIC, as opposed to only a camera matrix in the proposed scheme. Both schemes compute these entities N times, for an N-frame sequence. If the number of outliers is negligible, it can be shown that the computational requirements for solving a 4x9 and a 8x9 equation system is roughly half of that of a 11x12 one. However, as the ratio of outliers increase, the cost of computing the F-matrix dominate, since the number of trials in RANSAC drastically increase with the minimum sample set, as can be seen in Equation 3.13, making the GRIC-based metric computationally more expensive in practice. Therefore, the proposed pair selection method assesses more pairs by using a more informative metric and yet is computationally less expensive compared to the GRIC-based method.

3.10 Conclusion

In this chapter, basic modules of two-view and multiple-view reconstruction algorithms are discussed, as well as some enhancements, such as self-calibration and trajectory segmentation, to deal with metric reconstruction and dynamic scenes. Then the concept of frame pair ordering, or *prioritization*, is introduced, its niche is explained, and a metric for evaluating the utility of frame pairs for an

estimate-fusion type sequential algorithm is proposed. This metric is used to establish a processing order for *prioritized sequential 3D reconstruction* algorithm, the main contribution of this chapter. The proposed algorithm is shown to be competitive with the state-of-the-art through experiments. This result owes much to the frame ordering scheme, which is demonstrated to be superior to its most well-established opponent, GRIC-based ordering.

CHAPTER 4

EFFICIENT PIECEWISE PLANAR SCENE REPRESENTATION

The resulting sparse 3D point cloud produced by the algorithm described in the previous chapter can be considered as samples from the surface of the scene observed by the camera. Therefore, a straightforward method to achieve a dense reconstruction is to recover the surface via interpolation. In Section 1.1, the benefits of employing linear interpolation were pointed out, in the guise of piecewise planar representations, specifically, triangular mesh.

The sampling interpretation of the sparse reconstruction raises another issue: The sparse reconstruction algorithm takes samples of the scene wherever it can accurately compute them, leading to a sample set lying on an irregular grid. Moreover, it is quite possible that some parts of the scene are undersampled, while some others are oversampled. The former degrades the reconstruction quality, while the latter is a hazard to the efficiency of representation, an aspect that comes into play in the storage and transmission of the final scene representation. In order to overcome these problems, a sampling process that adapts the sampling density to the scene geometry is necessary.

The above discussion encapsulates the goal of this chapter: To design a dense 3D reconstruction algorithm capable of producing an accurate and efficient piecewise planar scene representation of a surface from its irregularly taken samples. In the following section, some design considerations for such an algorithm is stated as a starting point. Then, following the organization of the previous chapters, the basic tools, *Delaunay triangulation* for connecting the points into a mesh, or equivalently, linear interpolation, and frame rendering via homographies are introduced. These tools are used to build an algorithm that can achieve a rate-distortion efficient 3D representation of the scene [83]. Finally, the performance of the algorithm and its competitiveness are demonstrated through the experiments.

4.1 Design Considerations

In order to design an algorithm that yields an accurate piecewise planar representation with a relatively small number of planes, the following features are identified as desirable:

- **Piecewise planar reconstruction with triangular patches:** Piecewise planar representations offer a good approximation to many man-made and natural scenes. Besides, planes can be parameterized compactly, and give rise to special cases that provide significant computational savings [10]. Among all polygonal meshes that can be used to represent a piecewise planar surface, triangular meshes enjoy hardware support for their rendering, and an efficient representation by vertices only, when Delaunay triangulation is employed for their generation.
- **Coarse-to-fine operation:** The algorithms in the literature operate in a fine-to-coarse fashion [63][64][65], discarding vertices from a relatively dense point cloud as long as the distortion remains below a specified level.

The disadvantages of this approach are twofold: Firstly, increased number of vertices implies a more complex error surface for the problem, which can also be defined as finding a minimal set of vertices that describes a scene at a given distortion level. And secondly, it is inefficient to allocate computational resources on the estimation of entities to be discarded later. Operating in a coarse-to-fine fashion avoids both of these issues.

- **Feedback from representation to feature extraction:** Coarse-to-fine operation requires the use of a feedback path from representation back to feature extraction, to convey the directives of the representation block regarding to which parts of the structure requires refinement. This facilitates the use of an adaptive thresholding mechanism for feature extraction, resorting to less significant features only when more distinct ones fail to provide a sufficient reconstruction quality for a certain part of the representation.
- **Capability to update both structure and camera:** The methods in the literature assume that perfect camera matrices are available, and attribute any error to insufficient number of vertices, erroneous connections in the mesh [64] and sometimes to vertex positions [65]. However, within the context of 2D-to-3D uncalibrated video conversion, both the camera matrices and the point cloud are estimated from the input data, a fact that undermines the above assumptions. Hence, the algorithm should also be able to compensate for the errors in the camera parameters.
- **Projective operation:** Calibration errors are another common source of degradation. However, operating in a projective frame entirely eliminates them.

The problem focused in this chapter emphasizes relating the quality of the representation to its efficiency. Obviously, rate-distortion is an appropriate framework for studying this problem. In order to develop an algorithm in a rate-distortion framework, exact definitions of rate and distortion are necessary.

In this study, *rate* is defined as the number of vertices in the representation, a quantity that is strongly related to the number of bits to encode the representation itself. On the other hand, the choice of distortion metric is not straightforward. In the literature, the intensity error between a known image and its prediction is the most popular distortion metric, despite its oversensitivity to geometric errors. Moreover, minimization of an image-based error introduces a projective distortion to the structure estimate in case of erroneous camera matrices. The alternative is geometry-based error metrics, assessing how well the point cloud is modeled by the scene representation for a given vertex set. When accurate camera matrices are available, the minima of both of these metrics coincide. Otherwise, minimizing the image distortion transfers the error to the structure, or vice versa. This observation explains the popularity of PSNR in novel view synthesis and image prediction problems.

The operation of a rate-distortion efficient algorithm in the coding context, satisfying the above design specifications is depicted in Figure 4.1. As illustrated in the figure, the feature extraction and sparse structure estimation blocks compute a point cloud as a sparse representation of the scene, which is upgraded to a piecewise planar reconstruction in the representation module. Then, the compression block determines the locations, where more refinement is required,

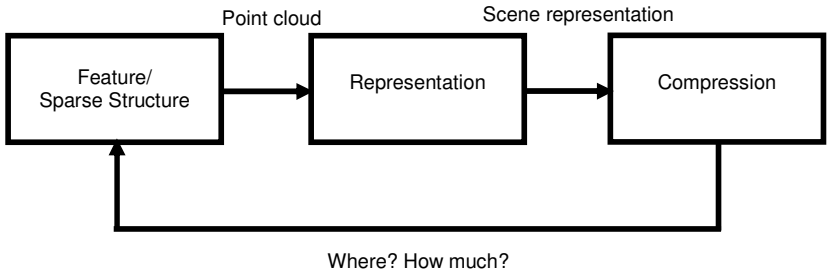


Figure 4.1: Piecewise planar 3D reconstruction in a rate-distortion framework.

and the bit budget states whether it is possible to realize this request. The feature extraction block extracts new features for these regions, and among this set of new features, the one in the least agreement with the current scene representation is added to the point cloud. The new point cloud is used to update and refine the representation. It should be noted the feedback operation facilitates the addition of new vertices in a way to improve the representation quality; therefore, the algorithm operates in a rate-distortion efficient fashion.

4.2 Surface Interpolation with Delaunay Triangulation

In this work, surface interpolation is performed indirectly in 2D. To this aim, first the 3D point cloud is projected onto the image plane of a reference camera. Then, the area bounding the projection is divided into triangles, by connecting the projected points (vertices) according to a certain rule. Finally, the triangulation is lifted to 3D by replacing the 2D vertices by the corresponding 3D points [77]. This method bestows the representation projective invariance, therefore, makes it possible to use a projective sparse reconstruction and camera set as input. However, it should be noted that the triangulation, when lifted to 3D, does not necessarily satisfy the rules enforced during its construction.

In order to partition the projection of the surface into planes, Delaunay triangulation is used. Delaunay triangulation employs the rule that no vertex can lie within the circumscribing circle of another triangle [77]. A Delaunay triangulation for a vertex set maximizes the minimum angle, therefore, gravitates towards equilateral triangles. This property leads to a "better looking" triangulation for many scenes [77]. Besides, a Delaunay triangulation is unique for a point cloud, i.e., it can be represented only by its vertices. The triangulation is implemented by using *Delaunay tree* [76], a data structure that allows dynamic updates of the connections between the vertices upon the addition of a new vertex,

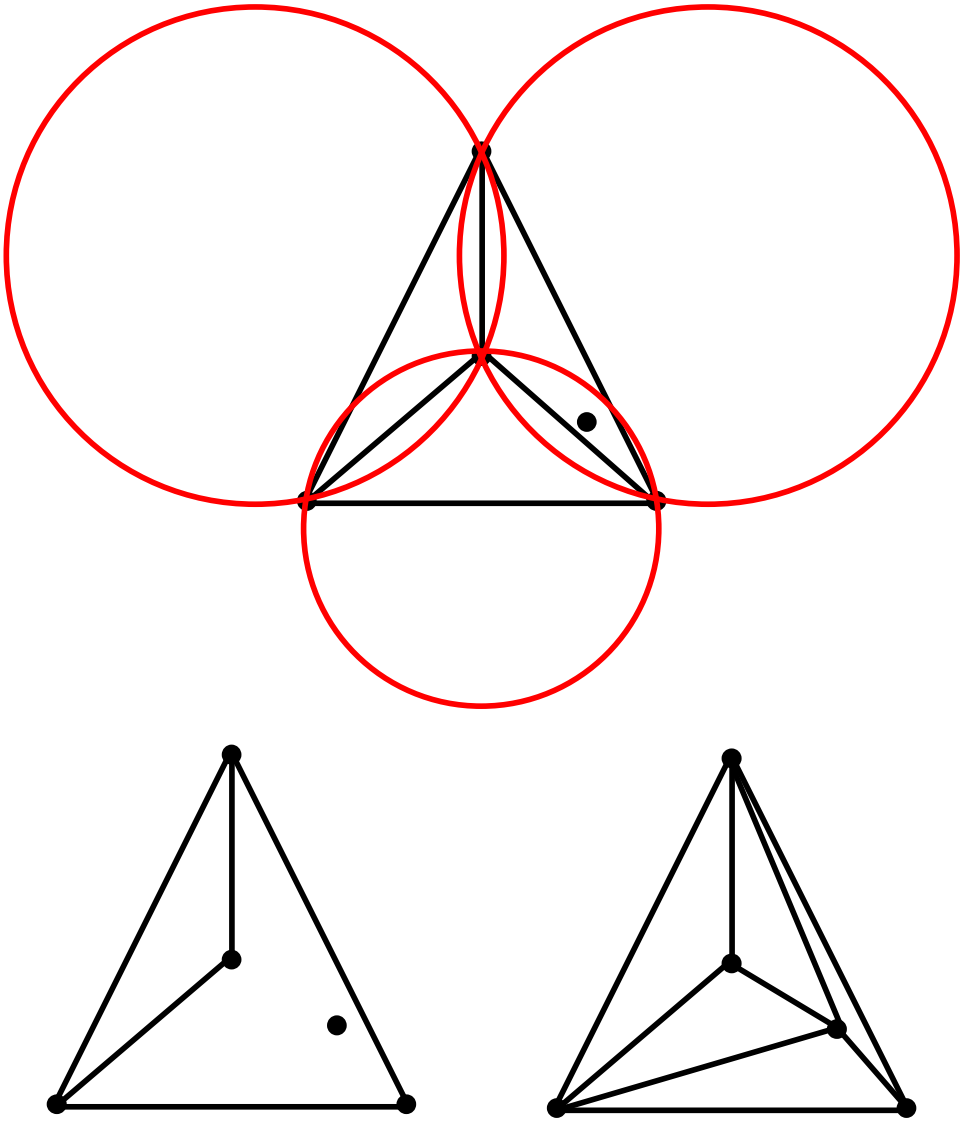


Figure 4.2: Incremental Delaunay triangulation. *Top:* New vertex, and the circumscribing circles of the triangles. *Bottom left:* Cavity formed by the removal of triangles in conflict with the new vertex. *Bottom right:* Triangulation after the insertion of the new node.

i.e., all modifications to the triangulation occur only where the new vertex violates the circumscribing circle rule. Below, a simplified version, one that does not include the removal of vertices, is described.

An incremental Delaunay triangulation algorithm is initialized by constructing a triangle bounding the entire vertex set. The algorithm inserts a new vertex into the triangulation by removing all triangles *in conflict* with the vertex, i.e., including the vertex in their circumscribing circles. This operation discards all common edges between the removed triangles. However, the edges shared by a removed and a surviving triangle are kept. These edges form a *cavity*, a polygon including the new vertex. The triangulation is updated by constructing triangles from the edges of the cavity and the new vertex. When the current triangulation is needed, the triangles related to the initial triangle should be removed. Figure 4.2 depicts a graphical illustration of the process.

A Delaunay tree is a structure that allows efficient identification of the triangles that are in conflict with the new node. Its nodes keep all generated triangles, both removed and surviving. A node is *killed* when the corresponding triangle is removed. Each node keeps the following information, in addition to the corresponding triangle and its circumscribing circle [76]:

- **Neighbors:** Neighbors of the triangle (at the instance of death, if killed).
- **Sons:** The triangles created from the edges of the node, when the node is killed.
- **Stepsons:** All dead and alive triangles sharing an edge with the node, from the inception of the node, until its death. For each edge, a separate list of stepsons is maintained.

Surviving nodes of the tree, or *leaf* nodes, correspond to the current triangulation.

In a Delaunay tree, the cavity is identified by traversing down all the nodes in conflict with the vertex. If a dead node is in conflict, this implies that some of its sons and/or stepsons are in conflict. Therefore, it is possible to descend down further by examining the offspring of the conflicting sons and stepsons recursively, until the leaf nodes in conflict are identified. The Delaunay tree speeds up the construction of the cavity by labeling some nodes as *not-in-conflict* with the new vertex; therefore, pruning the corresponding branch from the search tree. If the branches leading to a leaf through either son or stepson relations are pruned, the leaf is eliminated from the search process.

The insertion of a new vertex to the triangulation is summarized below [76]:

`Algorithm: Insertion of a New Vertex to a Delaunay Tree`

Input: Current Delaunay tree, new vertex.

Output: Updated Delaunay tree.

1. Add root node to the processing queue.
2. While the processing queue is not empty
    - ***a.*** If the current node is in conflict with the new vertex, and not yet visited
        - **i.** Mark the node as visited.
        - **ii.** Add sons and stepsons to the processing queue.
        - **iii.** If the current node is a leaf, kill the node.
            a. For all neighbors not in conflict with the vertex

i. Create a new node, son of the current node and stepson of the neighbor.

ii. Update neighborhood relationships.

3. Mark all nodes as not visited.

The center of the circumscribing circle, $(x_s;y_s)$ can be computed by

$$x_s = \frac{\det\left(\begin{bmatrix} x_1^2+y_1^2 & y_1 & 1 \\ x_2^2+y_2^2 & y_2 & 1 \\ x_3^2+y_3^2 & y_3 & 1 \end{bmatrix}\right)}{\det\left(\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix}\right)} \qquad y_s = \frac{\det\left(\begin{bmatrix} x_1 & x_1^2+y_1^2 & 1 \\ x_2 & x_2^2+y_2^2 & 1 \\ x_3 & x_3^2+y_3^2 & 1 \end{bmatrix}\right)}{\det\left(\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix}\right)}, \quad (4.1)$$

where $(x_i;y_i)$ are the coordinates of the i^{th} vertex of the triangle.

4.3 Rendering a View of the Scene

A view of the scene, as observed from a certain viewpoint, can be rendered by computing the intersection of the rays emanating from the corresponding camera center and passing through the pixels in the image, with the 3D scene representation. The texture at the intersection determines the intensity values for those pixels. When the texture information is contained in a texture image, the operation is equivalent to mapping the intensity values in the texture image to the view to be rendered.

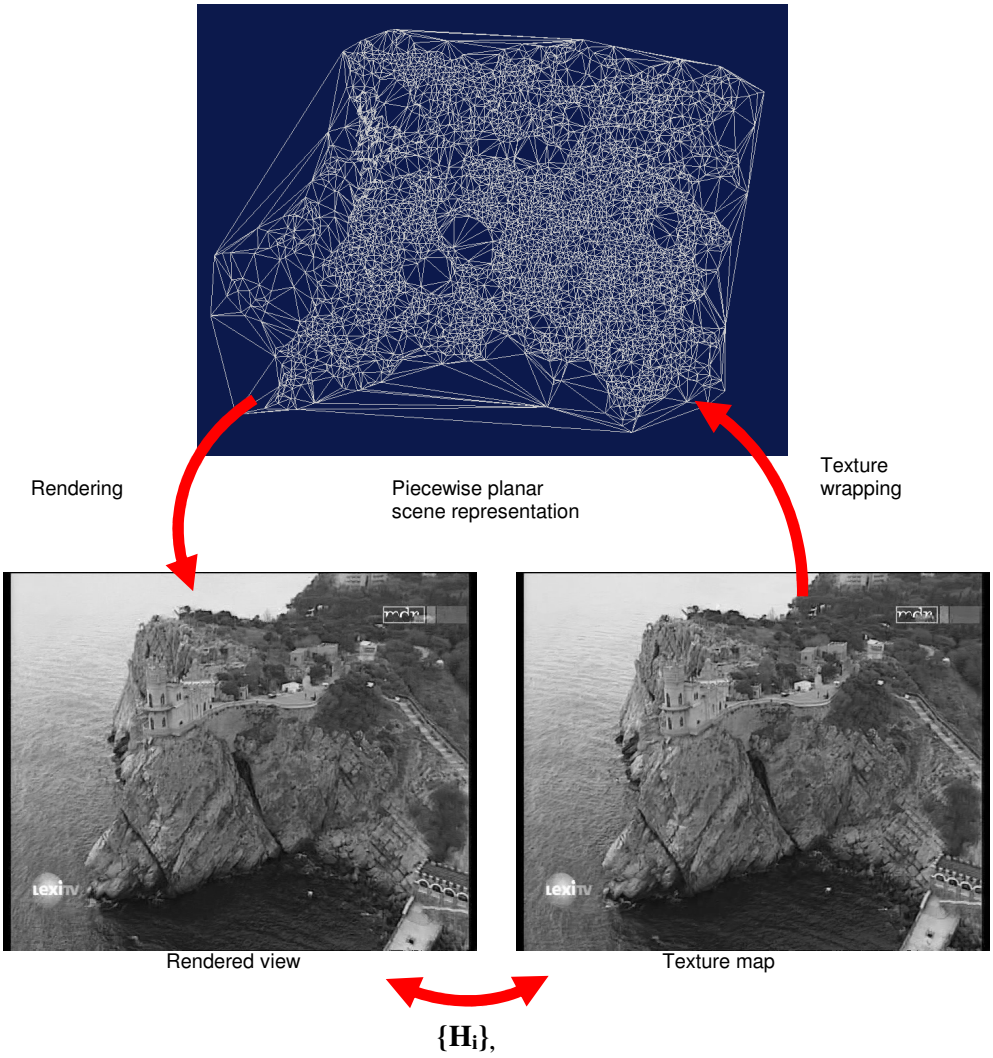


Figure 4.3: Frame rendering. Texture map contains the texture information for the scene, and mapped to it by wrapping it over the surface. $\{\mathbf{H}_i\}_i$, the collection of homographies can be used to render the frame seen at the viewpoint directly, instead of finding the intersection point of the rays from the viewpoint with the surface, and tracing a ray from it to the texture map.

In case of a piecewise planar scene representation, the knowledge of camera matrix associated with the texture image (henceforth it is referred as *reference camera and reference image*), allows the recovery of the mapping as a collection of homographies. This follows from the fact that images of a scene plane are related by a homography. Therefore, since images of a piecewise planar scene are composed of the images of the individual planar patches, the relation between the images is defined by a set of homographies, each of which is defined within the projections of the associated patch. The homography induced by the i^{th} scene plane equals to [10]

$$\mathbf{H_i} = \mathbf{P_2}\left(\mathbf{P_1^+} - \mathbf{C_1}\frac{\mathbf{n_i P_1^+}}{\mathbf{n_i C_1}} \right), \tag{4.2}$$

where $\mathbf{P_1}$ and $\mathbf{P_2}$ are the camera matrices belonging to the desired view and the reference image, respectively. The symbol "**+**" denotes pseudo-inverse, $\mathbf{C_1}$, the camera center of $\mathbf{P_1}$ and $\mathbf{n_i}$, the coefficients of the plane equation of the i^{th} scene plane. The process is illustrated in Figure 4.3.

When a ray intersects multiple scene planes, it becomes necessary to determine the first point of contact, to identify the homography to be used for the mapping. This amounts to determining which plane is visible at which pixel. The visibility is maintained by using a *Z-buffer*. Z-buffer keeps the depth of each scene point associated with a pixel in the view to be rendered [77]. When a new scene plane is rendered, a pixel within the projection of the patch is rendered via the corresponding homography, only if the plane is visible at that pixel, i.e., the value of the Z-buffer at that pixel is greater than the distance between the camera center and the point of intersection with the new plane. The distance between the camera center and a scene point is expressed as [10]

$$depth = \frac{\text{sgn}(\det \mathbf{M}) x_3}{X_4 \left\| \mathbf{m}^3 \right\|}, \tag{4.3}$$

where **M** is the first 3x3 submatrix of the camera matrix, and $\mathbf{m}^3$ is its third row. X_4 and x_3 are defined as

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \mathbf{P} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}, \tag{4.4}$$

It should be noted the order of intersections is projective invariant, due to the invariance of colinearity.

Below is a brief recapitulation of the view rendering algorithm.

`Algorithm: View Rendering`

Input: A list of triangular patches, reference image, reference camera matrix, camera matrix of the view to be rendered (viewpoint).

Output: A view of the scene, as observed from the viewpoint.

1. For each patch
    - ***a.*** Compute the corresponding homography.
    - ***b.*** For each pixel in the projection of the patch to the viewpoint
        - **i.** Compute the distance of the point of intersection to the camera center of the viewpoint.

ii. If the patch is visible, update the Z-buffer and transfer the pixel to the texture image, to find its intensity value.

4.4 Rate-Distortion Efficient Piecewise Planar Scene Reconstruction

Rate-distortion efficient piecewise planar scene reconstruction is a novel technique proposed in this dissertation, to build an efficient scene representation by the help of an algorithm in agreement with the design considerations laid out in Section 4.1. The algorithm refines an initial mesh, in such a way that the intensity error between a target image and its prediction from a reference image is minimized, through the addition of new vertices. The main operation cycle of the algorithm involves identifying the scene patch with the largest distortion, finding the vertex that violates the local planar model represented by the patch most, and adding that vertex to the representation to improve the local representation quality. This practice aims to obtain the maximum error reduction for each vertex introduced to the representation, therefore, yields a rate-distortion efficient representation. The loop terminates when the error converges, or the bit budget reserved for the representation is depleted. Subsequently, a non-linear optimization stage is employed to refine the results. The flowchart of the algorithm is depicted in Figure 4.4.

4.4.1 Sequential Phase

The algorithm can be minimally initialized with a target and a reference frame, from which a camera pair and an initial mesh can be estimated via the 2-view 3-D reconstruction technique of Section 3.2. However, if available, the algorithm is also capable of utilizing any initial estimates of the structure and camera matrices. The 2D-3D uncalibrated video conversion system proposed in this work, or stereo camera systems are examples of the cases, where such information is partially or

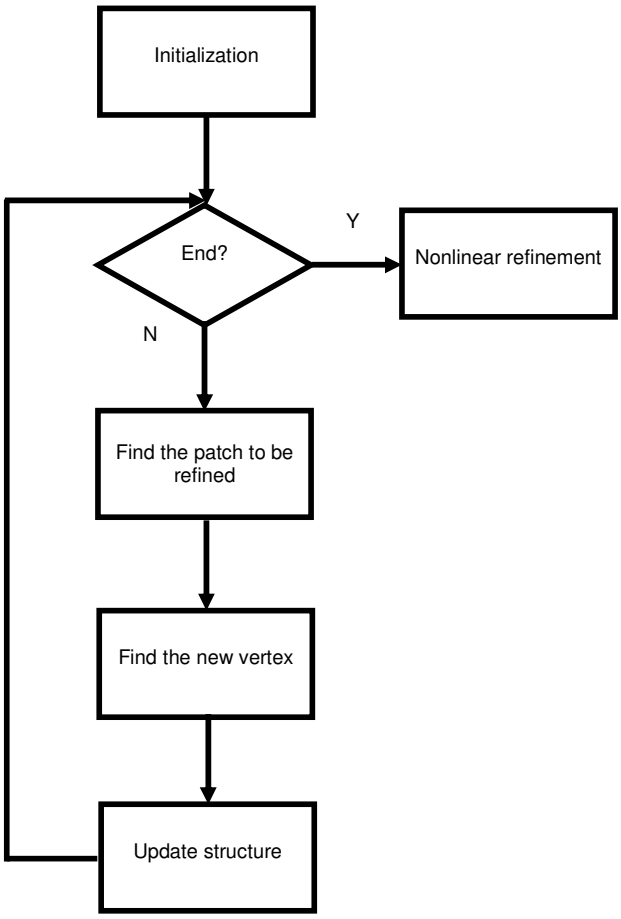


Figure 4.4: Flowchart of the rate-distortion efficient piecewise planar scene reconstruction algorithm.

fully available. The initial structure estimate is used to extract the bounding mesh (replacing the initial triangle in the triangulation algorithm of Section 4.2). Each cycle of the algorithm aims to find the vertex, whose addition to the representation yields the best improvement to the representation quality, measured by the sum of square of the difference between the target image and its prediction. This goal is achieved in two steps: First, the patch, whose projection corresponds to the region with the largest error, is determined. Then, the projections of the vertex are searched in this region, and its correspondence in the reference frame.

The prediction of the target frame is computed from the current scene representation and the reference frame, by using the view rendering algorithm described in Section 4.3. This prediction is segmented into regions, each of which corresponds to a patch in the scene representation. The patch, whose associated region has the largest total square error, is the one to be refined in the current cycle. The projections of the patch in both images are declared as the search regions for the next step.

In order to find the vertex, first, salient features in the search regions are extracted by any corner detector (e.g., Harris), and then correspondences are established through guided matching, discussed in Section 2.2. If no reliable matches are available, it is possible to repeat the feature extraction stage with a lower threshold, thus to resort to less reliable features, or to skip the patch, and continue with the next worst patch. Each feature pair defines a 3D point. The pair which has the least conformance to the local scene representation, i.e., the planar patch, is chosen to instantiate the new vertex. The conformance is measured via the *symmetric transfer error* [10], defined as

$$d = \left(\mathbf{x_1} - \mathbf{H^{-1}x}_2\right)^2 + \left(\mathbf{x_2} - \mathbf{Hx}_1\right)^2, \tag{4.5}$$

where $\mathbf{x_1}$ and $\mathbf{x_2}$ represent the corresponding features, and $\mathbf{H}$ is the homography induced by the planar patch, which relates the pixels in the reference and target frames in its corresponding region. If a 3D point is on the planar patch, Equation 4.5 should yield zero value for its projections.

The scene representation, i.e., the current 3D mesh, is updated with the new vertex, as described in Section 4.2. If the vertex indeed improves the representation quality, it is accepted; otherwise, it is rejected. The loop continues until the error converges, or the available bit rate is depleted.

The sequential stage of the algorithm is summarized below.

Algorithm: Rate-Distortion Efficient Piecewise Planar Scene Reconstruction- Sequential Phase

Input: A reference image, a target image, optionally, initial structure and camera matrices

Output: A piecewise planar representation of the scene

1. Until the prediction error converges or the bit budget is depleted
    - ***a.*** Determine the patch with the largest representation error.
    - ***b.*** Establish correspondences in the search regions associated with the patch.
    - ***c.*** Find the feature pair with least conformance to the patch.
    - ***d.*** Add the corresponding vertex to the representation.
    - ***e.*** Update the prediction of the target frame with the new planes.

4.4.2 Nonlinear Optimization

Following the termination of the sequential stage of the algorithm described above, the representation is refined via nonlinear optimization. The optimization procedure minimizes the total square error between the target image and its prediction, with respect to the vertices and camera parameters. The parameterization of the scene representation with vertices implies the assumption of a connected surface, i.e., a surface without any disjoint patches. The inclusion of camera parameters prevent the minimization procedure to introduce errors to the structure, to compensate for the inaccuracies in the camera matrices, a situation discussed in Section 4.1, when ground-truth camera matrices are not available.

The minimization problem is formally defined as

$$\min_{\mathbf{V},\mathbf{C}} \left(\sum_{x,y \in T} \left(T(\mathbf{V},\mathbf{C},x,y) - R(x,y) \right)^2 \right), \tag{4.6}$$

where T and R denotes the target and the reference frames, respectively, $\mathbf{V}$, vertices of the mesh and, $\mathbf{C}$, camera parameters. The dependency of T to $\mathbf{V}$ and $\mathbf{C}$ is explicitly indicated in the expression. The minimization procedure is subject to the constraint that the number of corresponding pixels in the target and reference images cannot be decreased more than a small fraction of its original value, to disallow the solutions that achieve a smaller cost by reducing the area of the target image that can be constructed from the reference image, through manipulations of $\mathbf{V}$ and $\mathbf{C}$.

In the preliminary experiments, an alternating scheme that minimizes the cost first with respect to vertices, and then, to camera matrices is observed to yield better

results, and thus is adopted. Depending on available computational resources, one of the two optimization schemes, *steepest descent* [68] or *simulated annealing* [78] can be employed to solve the minimization problem defined by Equation 4.6.

Steepest Descent

Steepest descent is a simple optimization algorithm for finding the minimum of a function. It is based on the observation that at a point, a function decreases fastest in the direction of negative gradient. Therefore, an iteration of the type

$$\mathbf{x_{n+1}} = \mathbf{x_n} - \lambda \nabla \mathbf{F}(\mathbf{x_n}) \tag{4.7}$$

is guaranteed to converge to a local minimum for a sufficiently small scalar λ. In Equation 4.7, $\mathbf{x_n}$ denotes the current solution, $\mathbf{x_{n+1}}$, the updated solution and $\nabla\mathbf{F}$, the gradient of the function [68].

Steepest descent algorithm requires the computation of the gradient of the cost function at the current solution. The gradient can be easily and efficiently computed by forward differencing, a numerical approximation method. Forward differencing involves perturbing each variable of the cost function with a small ε_i, to measure its effect on the cost. Each component of the gradient vector is approximated by dividing the difference in the cost function to the perturbation. A typical value for perturbation is max($|x_i 10^{-4}|$, 10^{-6}), for the i^{th} component of the solution vector [10].

Another important issue is the selection of step size, as it has a profound effect on the performance of the algorithm. A small λ leads to slow convergence, whereas a large λ might actually increase the error. An adaptive scheme, as employed in the proposed algorithm, increases λ when the error decreases, and vice versa.

Below is a summary of the steepest descent mechanism employed in the nonlinear minimization stage of the algorithm.

Algorithm: Nonlinear Minimization of Intensity Error via Steepest Descent

Input: Initial camera matrices and vertices, target image.

Output: Refined camera matrices and vertices.

1. Compute the gradients with respect to the camera parameters and vertex locations to initialize the step sizes.
2. Until the error converges or the maximum number of iterations reached
    - ***a.*** Compute the gradient with respect to the vertex locations.
    - ***b.*** Until the error decreases
        - **i.** Move in the negative direction of the gradient.
        - **ii.** If the error decreases, accept the movement and increase the step size for the vertices.
        - **iii.** Else decrease the step size.
    - ***c.*** Compute the gradient with respect to the camera parameters.
    - ***d.*** Until the error decreases
        - **i.** Move in the negative direction of the gradient.
        - **ii.** If the error decreases, accept the movement and increase the step size for the camera parameters.
        - **iii.** Else decrease the step size.

Simulated Annealing

Simulated annealing is a well-known stochastic optimization technique [78]. It operates by randomly sampling the solution space around the current solution. Its

distinguishing feature is, even if a solution increases the cost, there is a certain probability that it will be accepted. It is this property that makes it possible to avoid local minima, as the optimization procedure can actually climb uphill on the surface of the cost function, to get out of a basin around a local minimum- a feat steepest descent is not capable of. However, the price paid for this capability is a considerably slower convergence characteristic.

The probability of accepting an inferior solution is governed by a temperature parameter, which gradually decreases as the iterations proceed. A decreased temperature means a greedier operation, i.e., less likelihood to accept worse results. In this work, the probability is related to the temperature as [78]

$$p(f(\mathbf{x})) = \begin{cases} 1 & f(\mathbf{x}) < f(\mathbf{x}_{\mathbf{current}}) \\ \exp\left(\frac{1}{T}(f(\mathbf{x}_{\mathbf{current}}) - f(\mathbf{x}))\right) & else \end{cases}, \tag{4.8}$$

where f is the cost function, total square error between the target image and its prediction. $\mathbf{x}$ denotes the tested solution corresponding to a perturbation of, $\mathbf{x}_{\mathbf{current}}$, the current solution. If the tested solution is accepted, the current solution and T, the temperature parameter, is updated. The update rule for T is selected as [78]

$$T_k = \frac{T_{initial}}{k+1}, \tag{4.9}$$

where k is the update counter, and $T_{initial}$ is the initial value of T. Once a solution is accepted, it is further refined by the steepest descent algorithm, described in Section 4.4.1.

Four mechanisms are proposed to move the current solution in the solution space:

- **Move Vertex:** All vertex locations are perturbed randomly.
- **Move Camera:** Camera parameters are perturbed randomly.
- **Add Vertex:** A new vertex is randomly added to the reconstruction.
- **Remove Vertex:** A vertex is randomly removed from the reconstruction.

Each mechanism runs as a separate simulating annealing process, with its own temperature. Vertex move and camera move are run in batches of 10.

A summary of the algorithm is presented below.

Algorithm: Nonlinear Minimization of Intensity Error via Simulated Annealing

Input: Initial camera matrices and vertices, target image.

Output: Refined camera matrices and vertices.

1. Until the maximum number of iterations is reached
    - ***a.*** Randomly select one of "move vertex", "move camera", "add vertex-" or "remove vertex" actions.
    - ***b.*** Perturb the solution with the selected mechanism.
    - ***c.*** If the movement is accepted, update $\mathbf{x}_{\mathbf{current}}$, the current solution, and the temperature parameter of the selected mechanism.
    - ***d.*** Refine the solution with steepest descent.
    - ***e.*** If the new solution is the best solution so far, save the solution.

4.5 Experimental Results

The performance and the properties of the proposed algorithm were studied through extensive experiments. In the first set of experiments, the algorithm was run on synthetic and real data, to observe the convergence behavior and to explore the effects of incorrect vertex and camera parameter estimates. The second set of experiments was conducted to determine whether the solution was stable, by forcing the algorithm to follow different paths to convergence. In the third and fourth experiment sets, two design choices, coarse-to-fine operation and symmetric transfer error (STE) were justified by comparing their performance with their alternatives, fine-to-coarse operation and total square error (TSE). Finally, the rate-distortion efficiency of the representations produced by the proposed algorithm was compared with that of dense depth map representation, and block motion vectors.

4.5.1 Piecewise Planar Reconstruction Experiments

The piecewise planar reconstruction experiments were performed on the following data sets: *Struwwelpeter* is a synthetic data with ground-truth camera parameters and structure available. The imaged scene has 9 surfaces and 12 vertices. For *Venus* [79] and *Breakdancers* [80], only camera parameters are known. *Palace, Cliff* and *Wall* are acquired from TV broadcast, and *Flowerpot* from a converging stereoscopic camera set-up. Finally, *Wadham* and *Castle* belong to a collection of photographs of mostly planar scenes taken from various poses. The experiment results are presented in three parts, depending on the available ground-truth information.

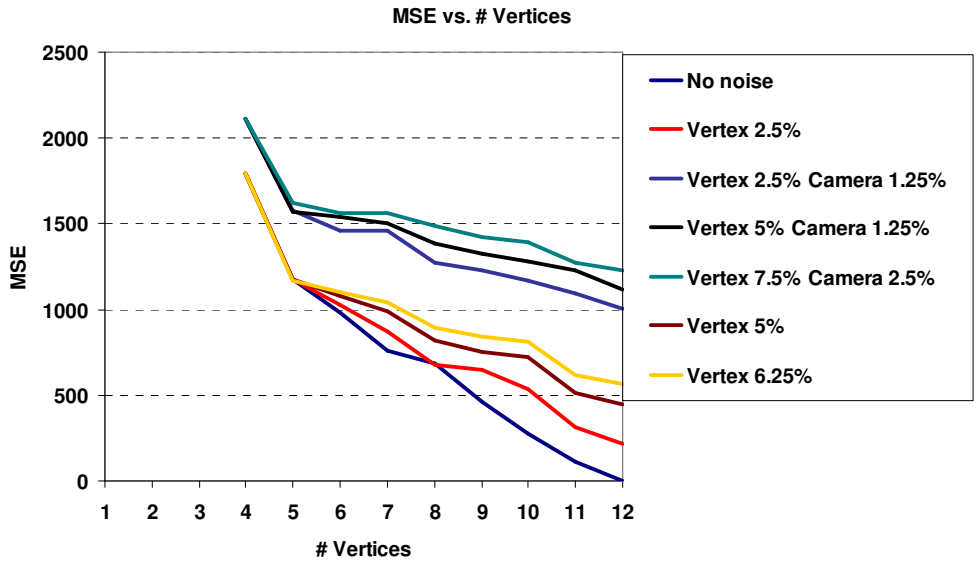


Figure 4.5: Sequential phase of *Cube* at various noise levels.

Case 1: Known Camera Parameters and Structure

In order to explore the effect of noise on the reconstruction process, the algorithm is run on *Struwwelpeter*, with varying levels of noise added to both ground truth camera parameters and vertices. The process illustrated in Figure 4.5, with a sample result for 7.5% noise on vertex positions and 2.5% noise on camera matrices depicted in Figure 4.6.

The experimental results indicate that the algorithm successfully recovers the correct structure in the absence of noise. However, the addition of noise significantly degrades the results. The effect is more pronounced especially when the camera parameters are affected by noise. However, as seen in Table 4.1 and Table 4.2, the final optimization phase achieves a remarkable error reduction. The reconstruction process is illustrated in Figure 4.5, with a sample result for 7.5% noise on vertices, and 2.5% on camera parameters.

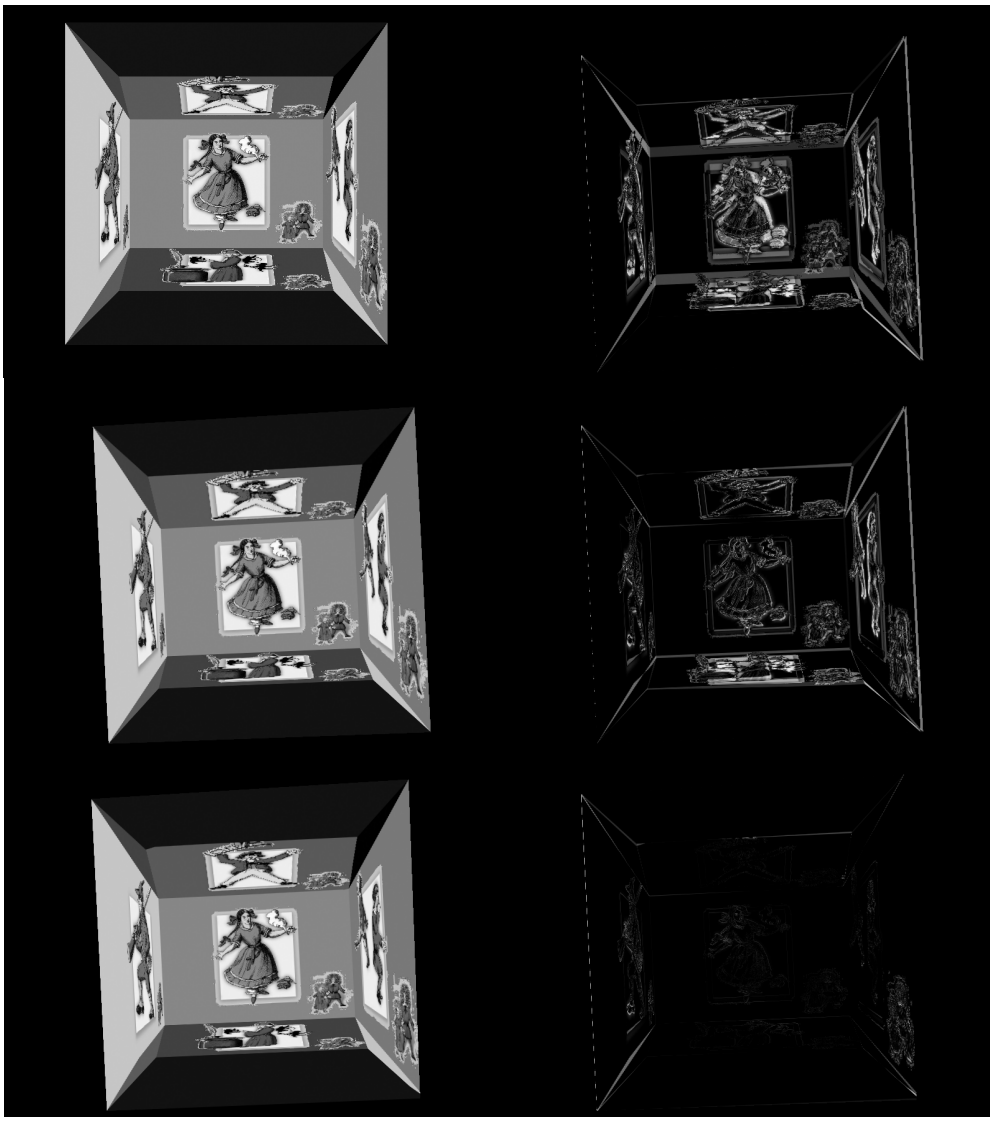


Figure 4.6: *Struwwelpeter*, with 7.5% perturbation on vertices and 2.5% perturbation on camera parameters. *Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear optimization, error after nonlinear optimization

Table 4.1: The performance of steepest descent in *Struwwelpeter*. "Sqn" and "SD" stand for the sequential phase and steepest descent

	MSE (Sqn/SD) (Equation 2.1)	**PSNR (dB)(Sqn/SD)**	**#Iterations**
Vertex 2.5%	216.69/ 12.03	24.77/ 37.33	20
Vertex 5%	448.68/ 24.26	21.61/ 34.28	59
Vertex 2.5% Camera 1.25%	1003.09/ 97.16	18.12/ 28.26	68
Vertex 7.5% Camera 2.5%	1231.36/ 125.51	17.23/ 27.16	100

Table 4.2: The performance of simulated annealing in *Struwwelpeter*. "Sqn" and "SA" stand for the sequential phase and simulated annealing.

	MSE(Sqn/SA) (Equation 2.1)	**PSNR (dB)(Sqn/SA)**	**#Iterations**
Vertex 2.5%	216.69/1.63	24.77/ 46.01	82
Vertex 5%	448.68/ 24.26	21.61/ 34.28	1
Vertex 2.5% Camera 1.25%	1003.09/ 49.23	18.12/ 31.20	651
Vertex 7.5% Camera 2.5%	1231.36/ 124.99	17.23/ 27.16	46

Case 2: Known Camera Parameters- Unknown Structure

Another set of experiments was conducted on *Venus* and *Breakdancers.* For *Breakdancers*, the ground-truth camera matrices were available, and for *Venus*, the only available information was that the frames belonged to a horizontally panning camera. However, this a priori knowledge was sufficient to compute an exact projective camera matrix. The results are presented in Figures 4.7, 4.8, 4.9, 4.10, and Tables 4.3 and 4.4. The black regions in the predicted images are the parts missing in the final representation due to the lack of feature points.

The most notable observation highlighted by the experiments is the number of vertices at which the error converges. In both *Venus* and *Breakdancers*, roughly 30 vertices are enough to obtain a PSNR beyond 32 and 30dB, respectively. Moreover, the results are improved in excess of 20%, via simulated annealing. Steepest descent provided a modest improvement when compared to the simulated annealing; however, it should be noted that the latter involves employing steepest descent at promising locations of the solution space, therefore explores a larger tract of the solution space.

The relatively early convergence and the poorer performance of the steepest descent, and non-linear optimization phase in general, in comparison to the case of known camera parameters and structure can be attributed to three factors. Firstly, the errors in the localization and matching of the features limit the performance. Secondly, more vertices usually yield a more complex error surface, with an increased number of local minima. Relative success of simulated annealing is actually a testament to its ability negotiate with such error surfaces better. And finally, another source of error is the discrepancy between the scene model, a connected surface, and the disconnected planes in the scene, such as the breakdancer at the centre in Figure 4.10.

Table 4.3: The performance of steepest descent in *Venus* and *Breakdancers*. "Sqn" and "SD" stands for the sequential phase and steepest descent

	MSE (Sqn/SD) (Equation 2.1)	PSNR (dB)(Sqn/SD)	#Iterations
Venus	34.72/ 30.44	32.73/ 33.30	3
Breakdancers	61.00/ 53.76	30.27/ 30.83	10

Table 4.4: The performance of simulated annealing in *Venus* and *Breakdancers*. "Sqn" and "SD" stands for the sequential phase and simulated annealing.

	MSE(Sqn/SA) (Equation 2.1)	PSNR (dB)(Sqn/SA)	#Iterations
Venus	34.72/ 29.38	32.73/ 33.45	9
Breakdancers	61.00/ 43.46	30.27/ 31.75	62

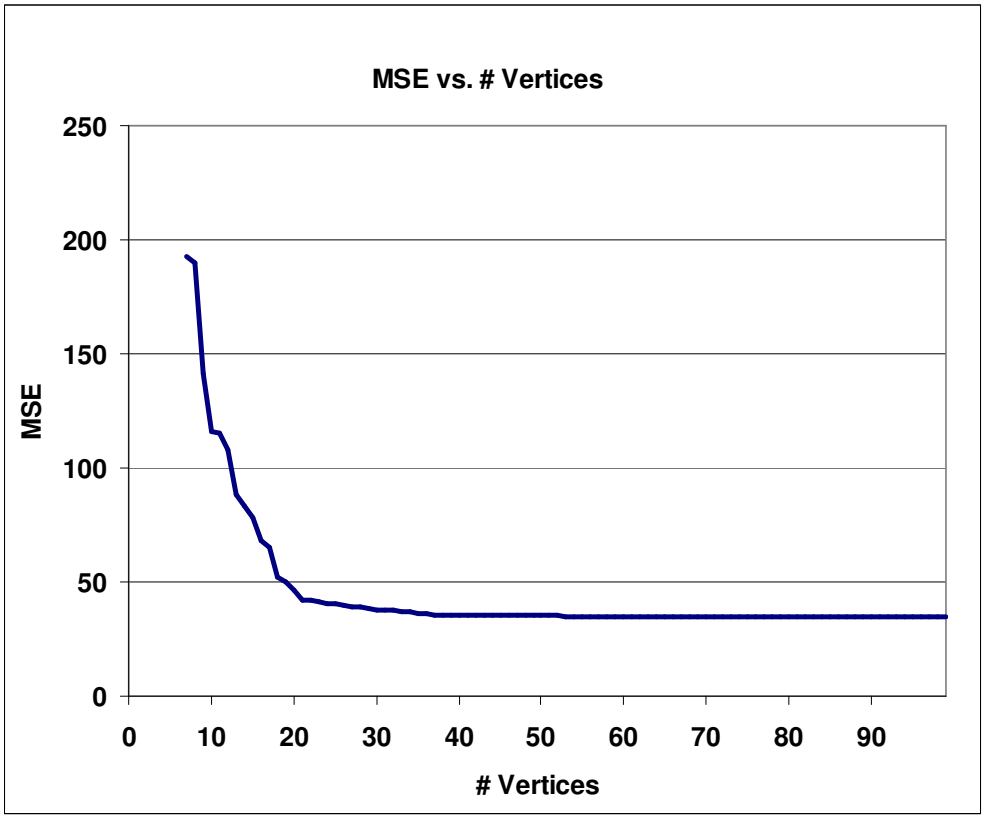


Figure 4.7: Sequential phase of *Venus*.

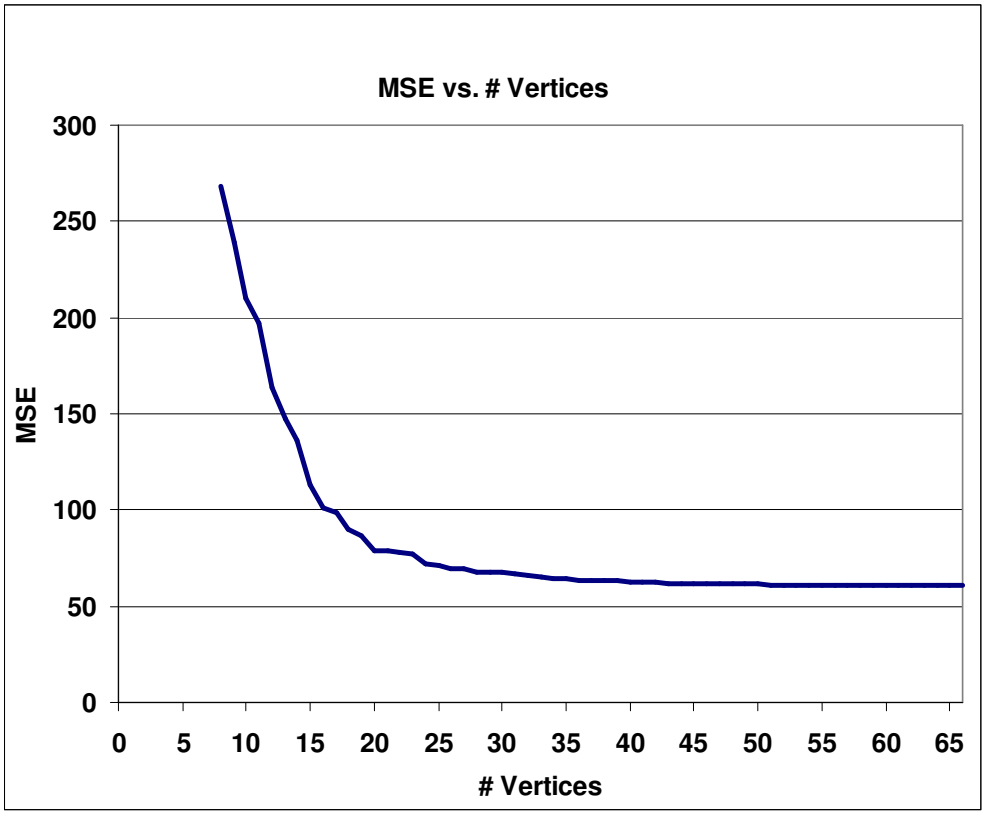


Figure 4.8: Sequential phase of *Breakdancers*.

Figure 4.9: *Venus. Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

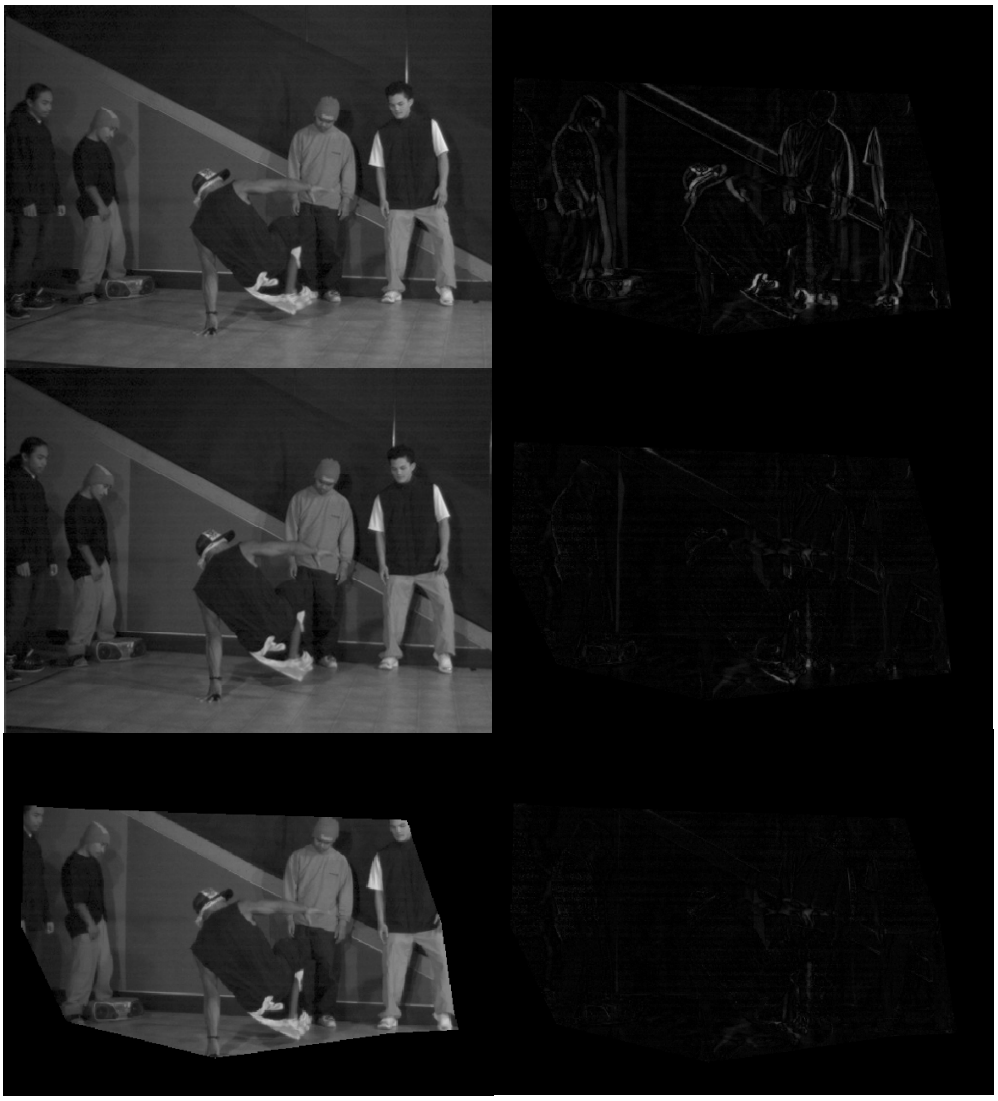


Figure 4.10: *Breakdancers. . Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

Case 3: Unknown Camera Parameters and Structure

The unknown camera and structure case corresponds to the problem considered in this work, 2D to 3D conversion from uncalibrated video sequences, therefore was analyzed in more detail than the above cases, with experiments on *Palace, Cliff, Wall, Castle [96], Wadham [98]* and *Flowerpot [97]*. In MSE calculations, the regions occupied by logos (e.g., lower right corner in *Flowerpot* and lower left corner in *Wall*), were excluded. Moreover, in *Castle*, the portions of the image including the trees were edited out, as their contribution to the error was large enough to disrupt the experiment aimed at studying the performance of the algorithm in a mostly-planar scene. In all experiments, both the camera parameters and the structure were estimated directly from the input frame pairs. The experiment results are presented in Figures 4.11-22.

In all experiments, the graphs depicting the progress of the sequential phase shows a similar, and the intended, behavior of convergence for the distortion, as the number of vertices increases. However, the convergence occurs at different number of vertices and to different error levels: In *Palace, Cliff*, *Castle* and *Wadham*, convergence is reached at around 50 vertices, while for *Flowerpot*, the convergence point is encountered at as late as 100 vertices. Moreover, the success of non-linear optimization phase also varies considerably, as presented in Table 4.5 and Table 4.6. These observations can be attributed to several causes, each of which is discussed below.

The most obvious source of error is the inaccuracies in camera parameters and vertex positions, as evidenced by the upwards trend in the final errors from known camera parameters and structure case to unknown camera parameters and structure case. It is also the best addressed one, as the sequential phase attempts to eliminate the unreliable vertices, and the non-linear optimization stage tries to combat against these errors, by displacing the vertices and the camera parameters to obtain a better solution.

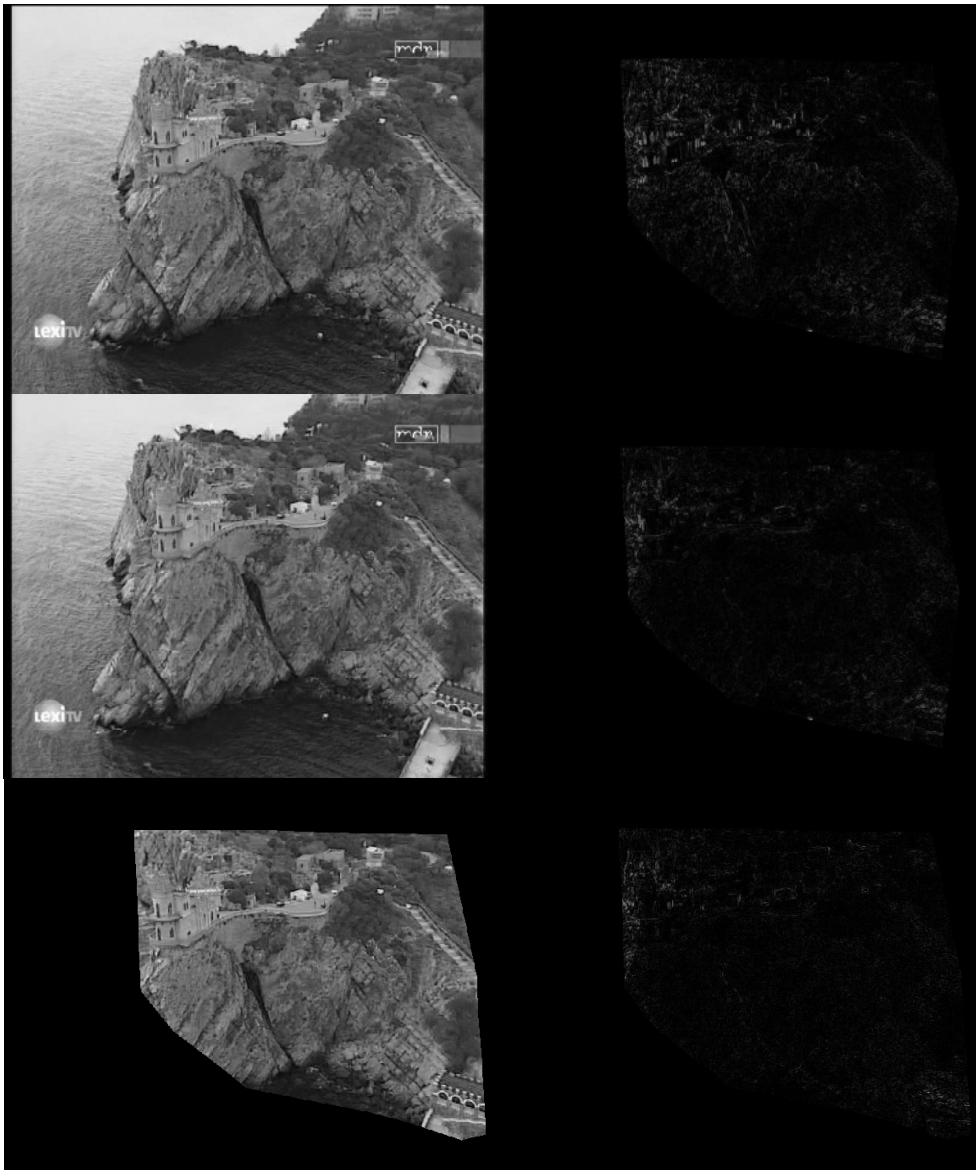


Figure 4.11: *Cliff. . Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

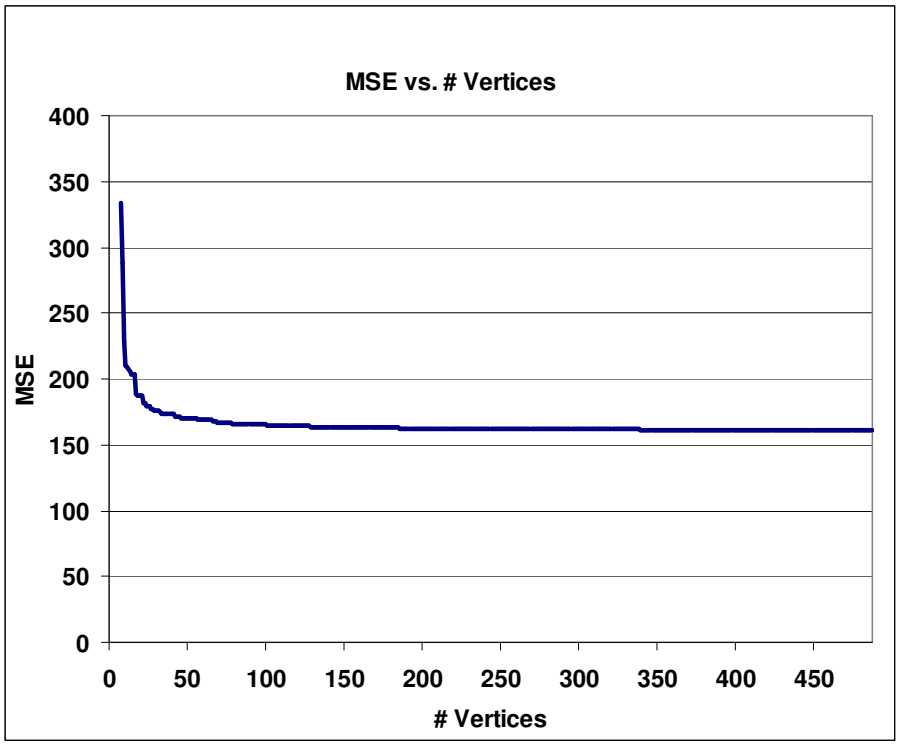


Figure4.12: Sequential phase of *Cliff*.

Figure 4.13: *Wadham*. *Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

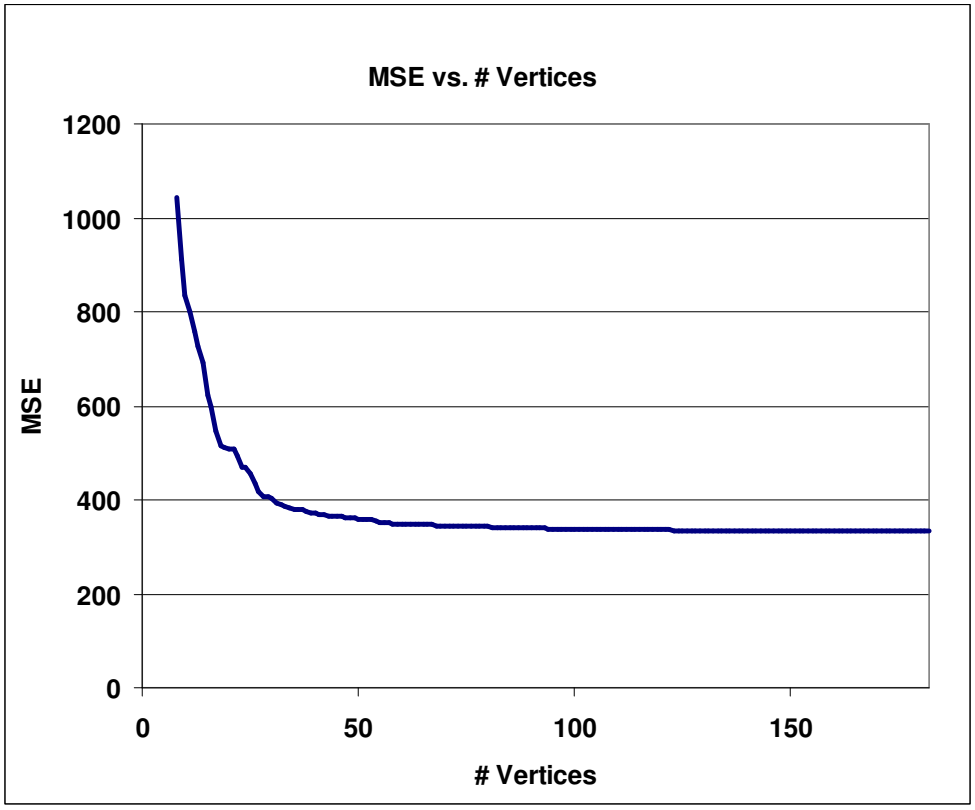


Figure 4.14: Sequential phase of *Wadham.*

Figure 4.15: *Castle*. *Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

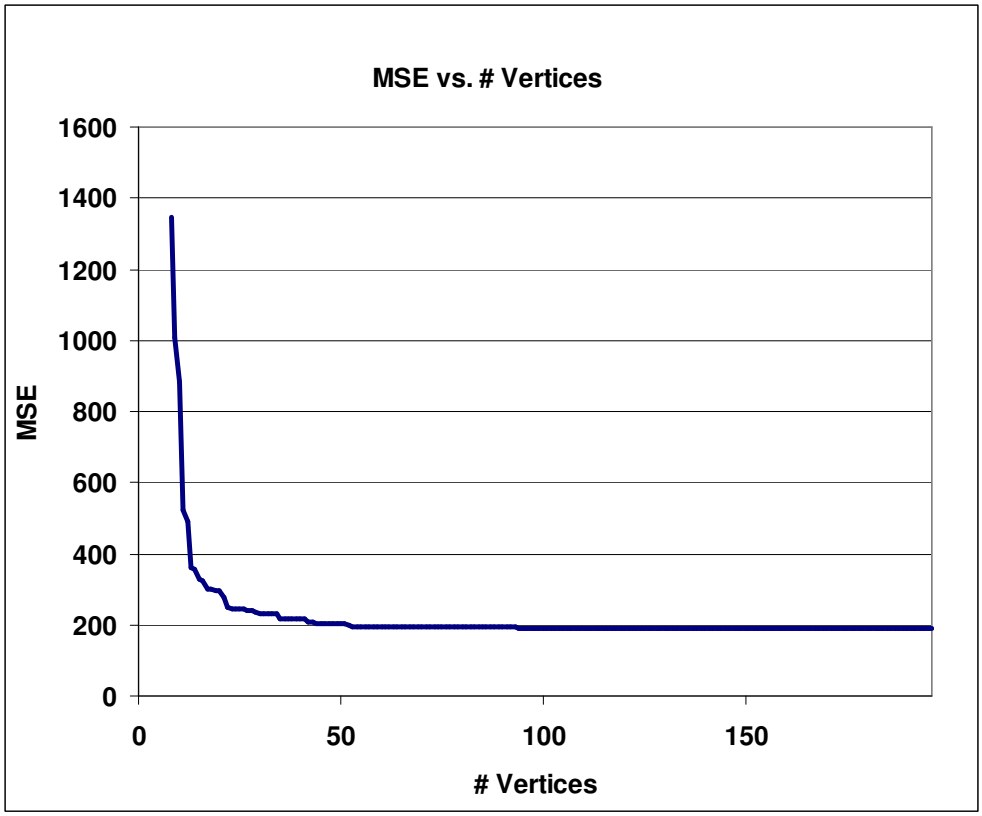


Figure 4.16: Sequential phase of *Castle*.

Figure 4.17: *Flowerpot*. *Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

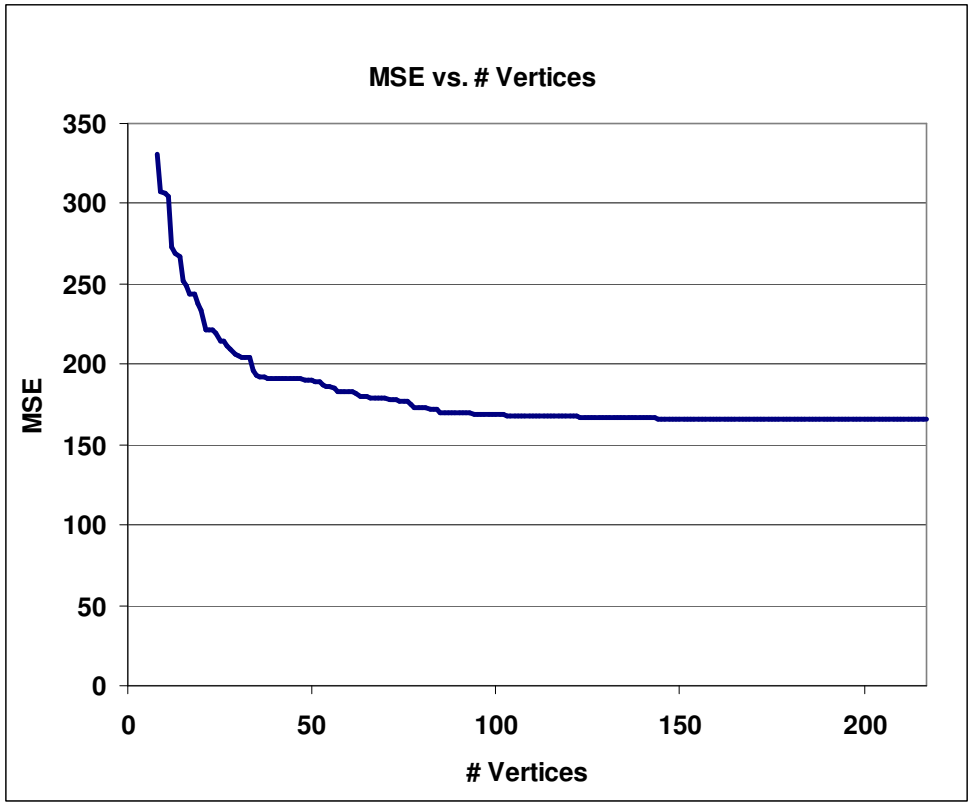


Figure 4.18: Sequential phase of *Flowerpot.*

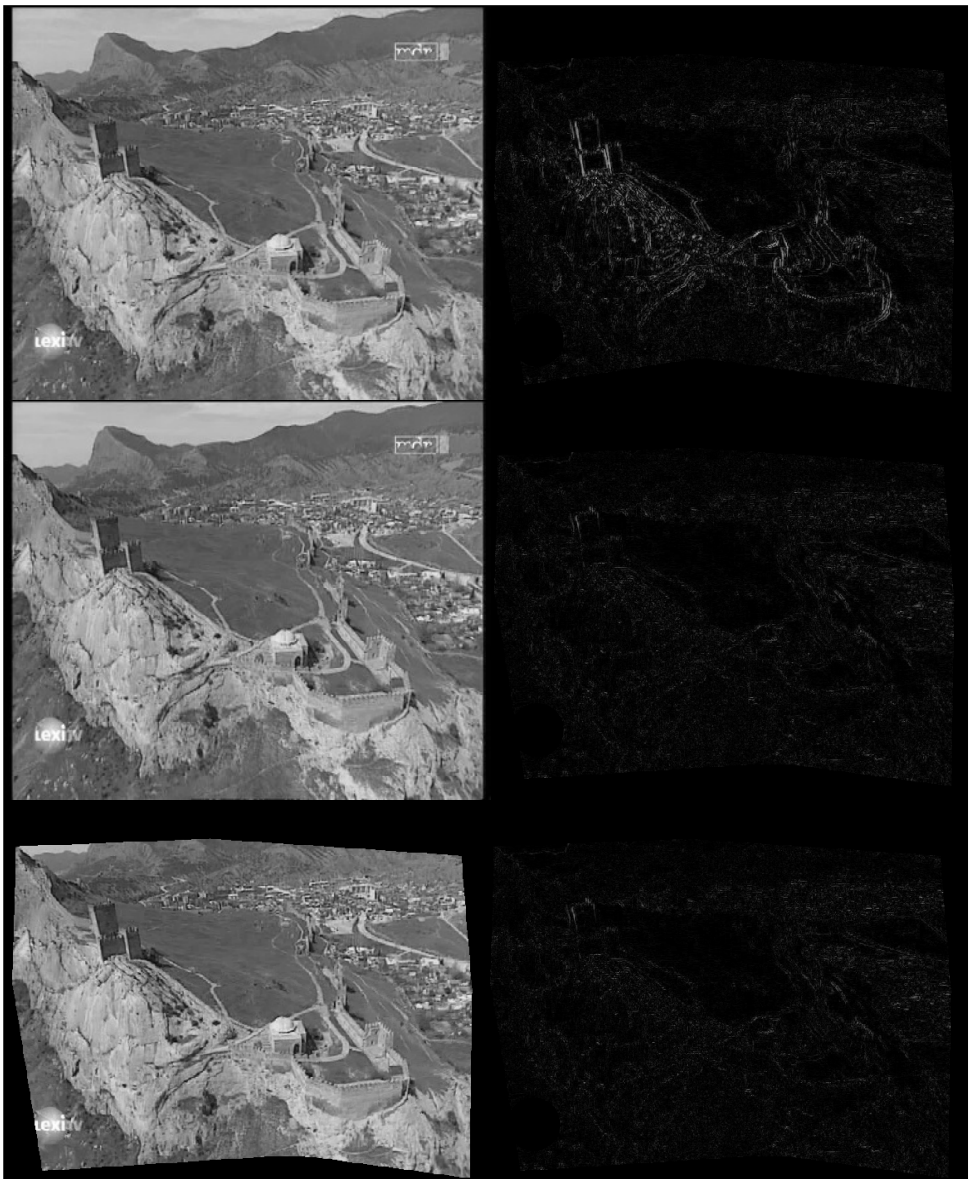


Figure 4.19: *Wall Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

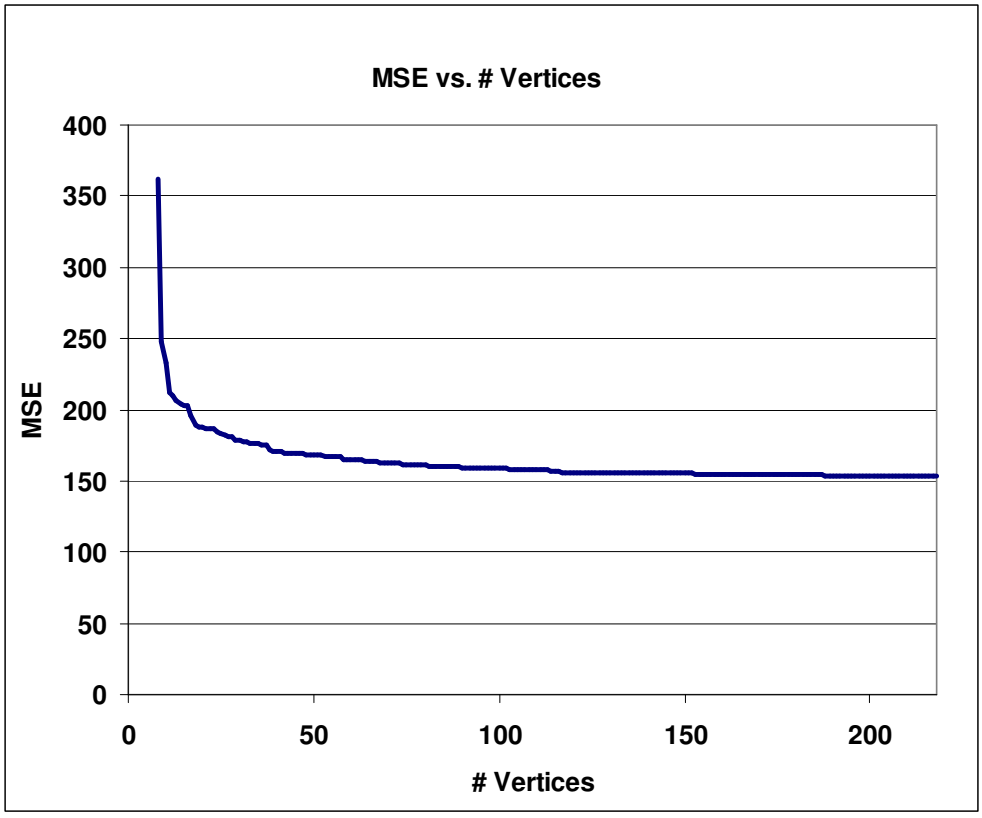


Figure 4.20: Sequential phase of *Wall.*

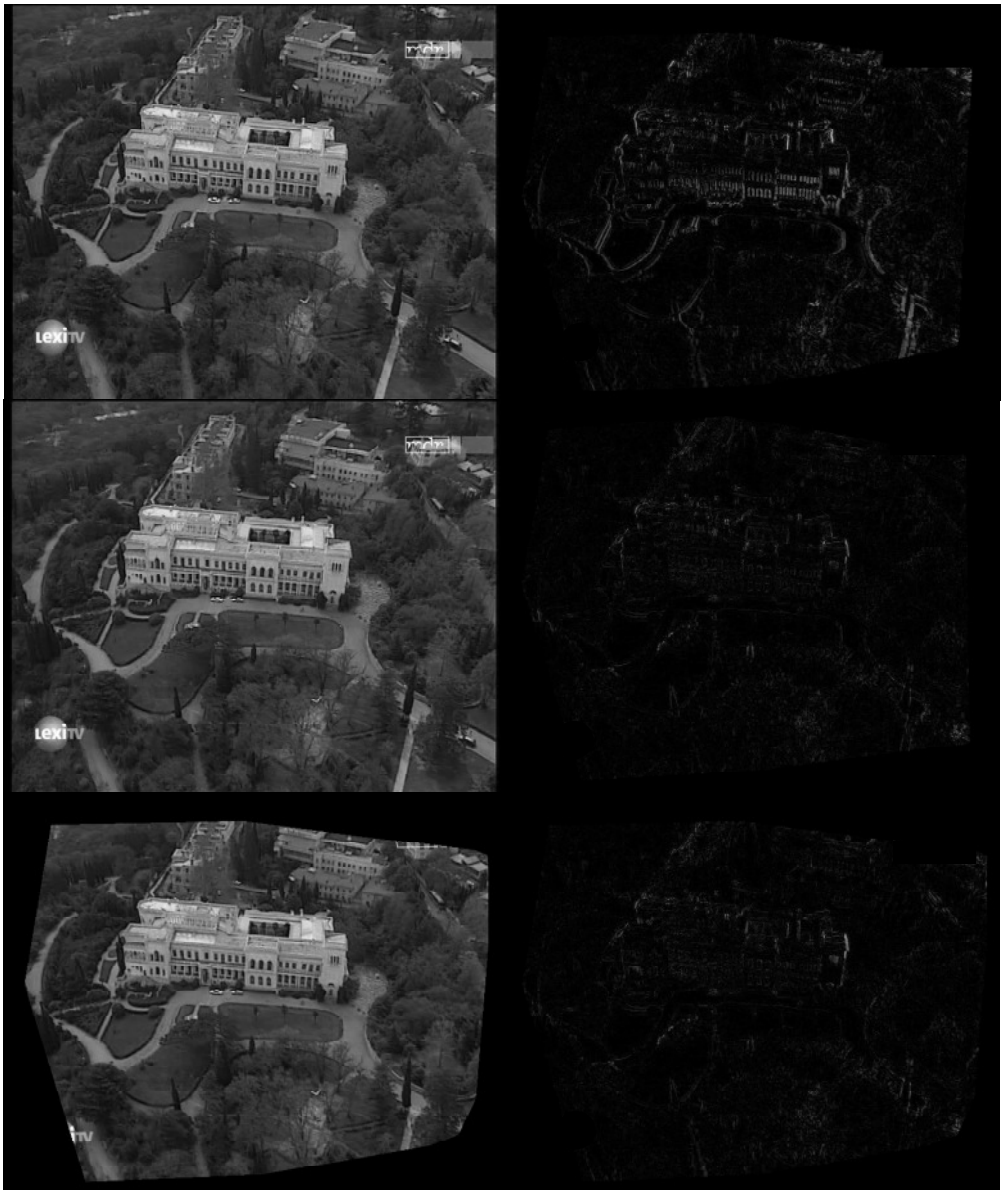


Figure 4.21: *Palace*. . *Left column, top to bottom:* Reference, target and predicted frames. *Right column, top to bottom:* Initial error, error before nonlinear stage, error after nonlinear stage.

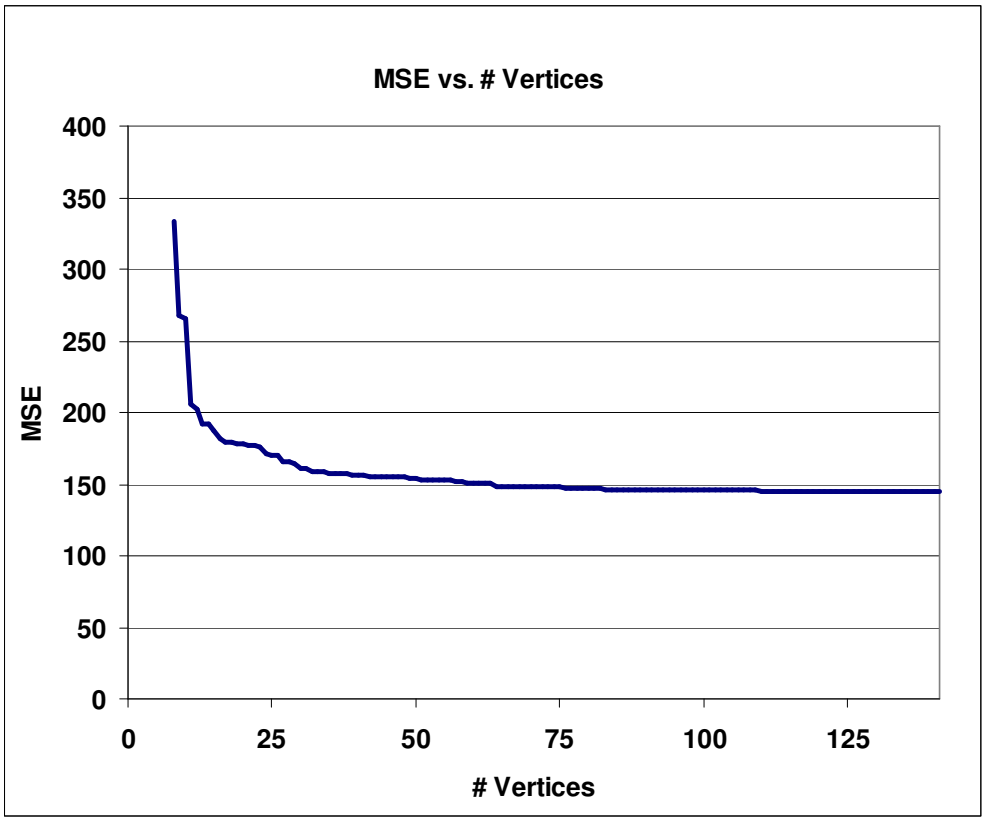


Figure4.22: Sequential phase of *Palace*.

Table 4.5: The performance of steepest descent. "Sqn" and "SD" stand for the sequential phase and steepest descent

	MSE (Sqn/SD) (Equation 2.1)	PSNR (dB)(Sqn/SD)	#Iterations
Cliff	161.55/ 138.08	26.05/ 26.73	5
Wall	153.46/ 141.09	26.27/26.64	4
Wadham	332.38/293.82	22.91/23.45	14
Castle	187.41/175.82	25.40/25.68	6
Flowerpot	165.84/155.46	25.93/26.21	5
Palace	145.02/134.28	26.51/26.85	3

Table 4.6: The performance of simulated annealing. "Sqn" and "SA" stand for the sequential phase and simulated annealing.

	MSE (Sqn/SA) (Equation 2.1)	PSNR (dB)(Sqn/SA)	#Iterations
Cliff	161.55/ 119.51	26.05/ 27.36	252
Wall	153.46/ 141.09	26.27/26.64	1
Wadham	332.38/264.49	22.91/23.91	83
Castle	187.41/175.82	25.40/25.41	1
Flowerpot	165.84/155.46	25.93/26.21	1
Palace	145.02/133.22	26.51/26.89	168

Another source of error follows from the violation of one of the fundamental assumptions of the algorithm, that the intensity values of the target image can be computed from those of the reference image perfectly, given the correct geometry and camera matrices, thus the minimization of MSE leads to better estimates of these parameters. The assumption generally holds for parts of the scene with low intensity variation, such as distant or flat portions of the image. However, when the frames are taken from considerably different positions, occlusions and disocclusions reduce its validity, as seen in Figure 4.22. Moreover, reflecting surfaces, such as windows in Figure 4.23, may also cause this assumption to break. Finally, the changes in the illumination of the scene weaken the relation between the reference and the target image. The mean intensity values of the images, along with MSE values were presented in Table 4.7 to demonstrate the link between the illumination and the distortion. The limitations of the predictability of the target image from the reference image causes a larger residual error and create local minima that cannot be avoided by the proposed algorithm, as it is not equipped with any tools to deal with errors not caused by camera and geometry parameters, except for significant structure deformations that are likely to increase the error. This observation also highlights the inadequacy of image-based error metrics, specifically, illumination-variant ones such as MSE or PSNR, to drive a scene representation process. The impact of this phenomenon is best observed in *Castle* and *Wadham*, as in both experiments, the sequential phase starts from a rather high MSE, and the residual error is highest among all results.

One final major source of error is incorrect connections between the vertices, leading to the generation of planes non-existent in the scene. These planes model the local structure erroneously, introducing effects, such as bending, as illustrated in Figure 4.25. Two major causes for such erroneous model are identified as mesh construction in 2D, and missing vertices. The former error source is the price paid for projective invariance: Since the mesh is constructed in 2D, it is possible that two far-away and unrelated vertices in 3D might be projected to close locations. In that case, these points are connected, forming one edge of a plane non-existent

Table 4.7: The relation between the mean intensity differences of the target and reference frame and the distortion.

	Reference Mean	Target Mean	MSE (Equation 2.1)
Wall	120.77	121.46	141.09
Wadham	113.10	111.51	264.49
Castle	107.68	114.63	186.94
Flowerpot	89.61	90.36	155.46
Cliff	99.72	98.60	119.51
Palace	66.57	66.67	133.22

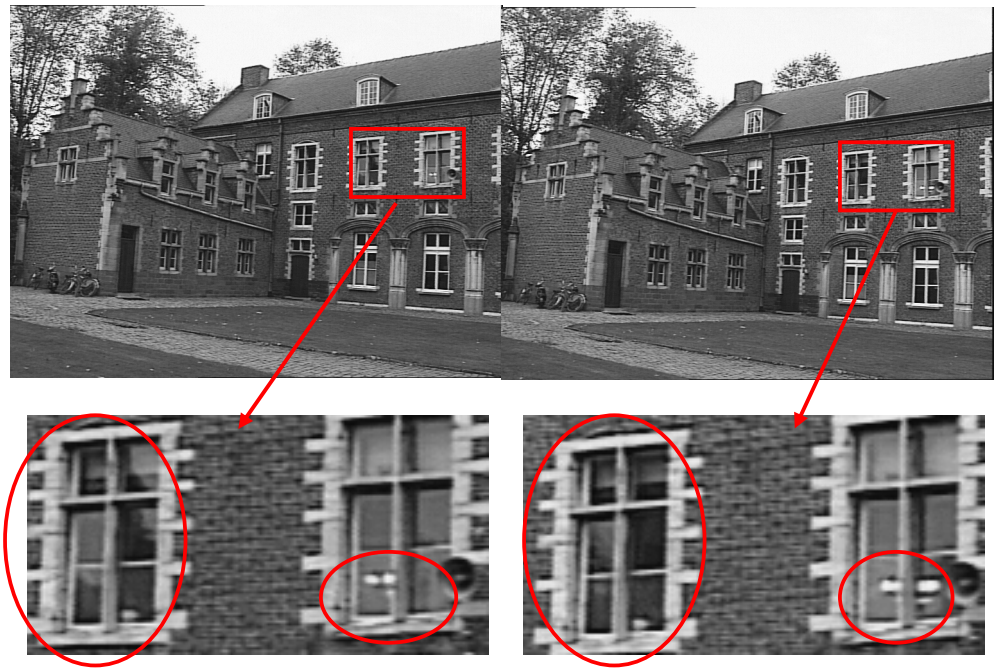


Figure 4.23: Detail from *Castle*. The regions violating intensity predictibility assumption are marked with red squares in the top row, and, enlarged in bottom row.

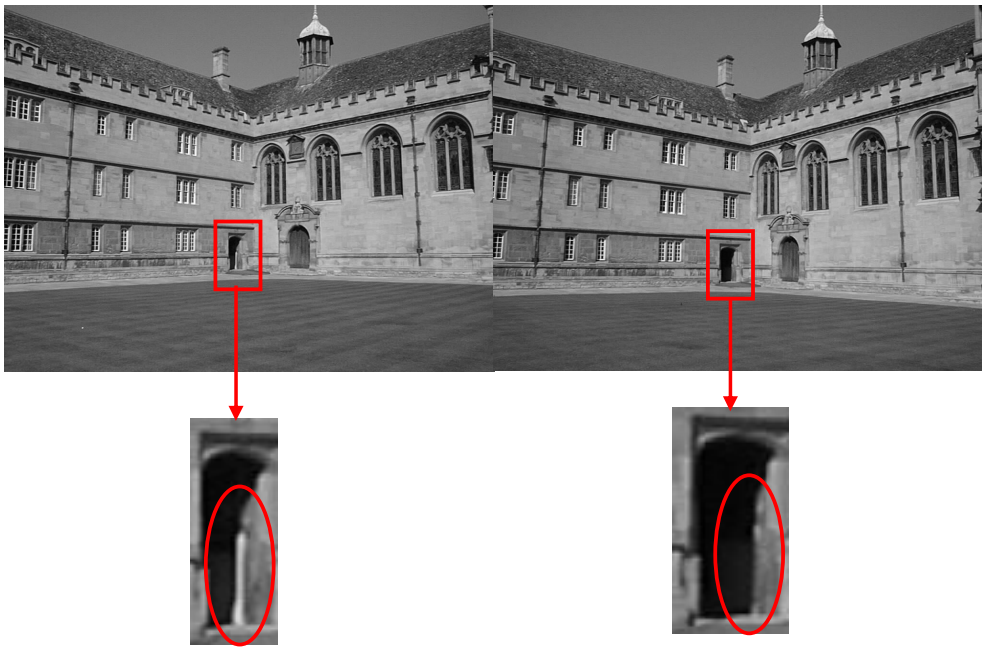


Figure 4.24: Detail from *Wadham*. The regions violating constant intensity assumption are marked with red squares in the top row, and, enlarged in bottom row.

in the scene. The latter error source is the product of the limitations of feature extraction and matching procedures, as it is not always possible to recover a corresponding feature pair set both sufficiently dense enough to describe the boundary of the scene planes correctly, and accurate enough for reliable 3D position estimation. The lack of features results in estimated plane boundaries not coinciding with the actual scene boundaries, and vertices belonging to the interior of a plane being connected to vertices of other planes, instantiating erroneous local planar models.

In an attempt to quantify the effect of meshing errors, an experiment was conducted, in which a suitable subset of feature pairs was selected manually to ensure a correct mesh. When processing the interactively selected feature pairs, the feature rejection mechanism was not utilized to make sure that all feature pairs were used in the mesh, in spite of the occasional increases in the error. The

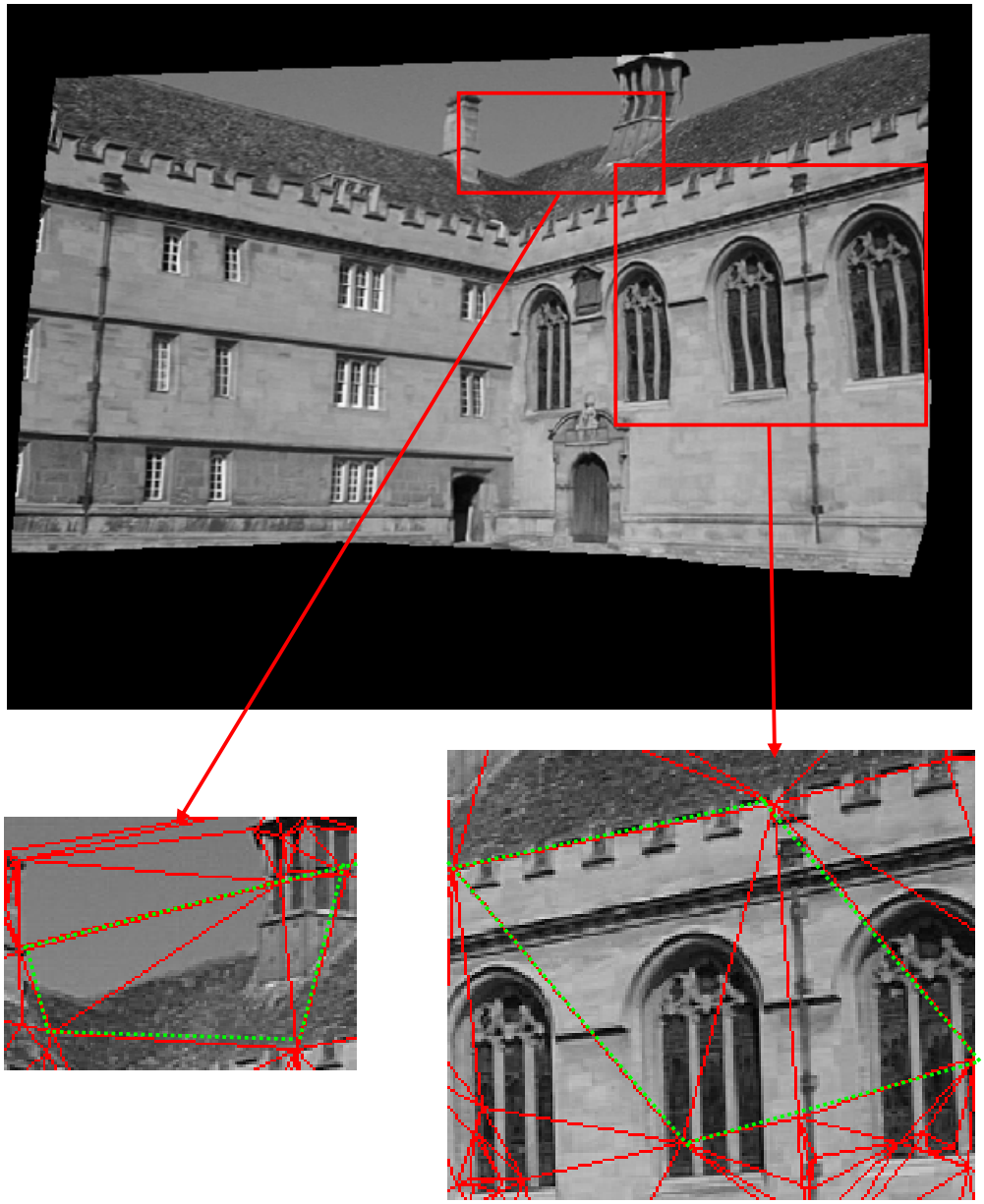


Figure 4.25: Structure deformations in *Wadham*. Deformed parts are marked with red squares in the top row, and enlarged in the bottom row. Red (solid) triangles indicate the underlying mesh, and green (dashed) lines delineate the plane collections with erroneous normals, causing structure deformations.

experiment results, depicted in Figure 4.27 and 4.29, indicate a remarkably faster convergence rate, when a mesh better conforming to the scene planes is used. Figures 4.26 and 4.28 illustrate these meshes at the point of convergence. These figures support the expectations that a mesh in agreement with the scene structure improves the performance. The results imply that the proposed algorithm can benefit considerably from the incorporation of any information about the actual plane boundaries, recovered from a line-detection or segmentation module, as apparently, corners alone do not always sufficiently describe these entities.

A final observation that deserves a mention is the performance of the non-linear optimization tools. Steepest descent can only modify the camera and vertex location parameters; therefore, the improvement it can provide is limited to what can be achieved through these tools. However, as discussed above, these are not the only mechanisms that affect the distortion. As for simulated annealing, as in all stochastic optimization algorithms, if the previous stage achieves a deep and stable minimum in the cost function, any further improvement comes at a high computational cost. In *Cliff* and *Wadham,* simulated annealing provides a notable improvement in distortion; however, in other cases, the computational cost proves to be prohibitive. The situation probably follows from the fact that both simulated annealing and steepest descent algorithms share similar tools, so the simulated annealing generally explores the other local minima in the vicinity of the minimum reached by the steepest descent by perturbing the vertices and camera parameters, for a given vertex set. If the steepest descent stage obtains a minimum considerably deeper than any other minima in its vicinity, the only mechanism the simulated annealing possesses to escape and to explore other regions of the solution space is the vertex deletion/addition mechanism, as it can indirectly alter the connections, hence the topology of the error surface with respect to vertices and camera parameters. However, an optimization procedure equipped with more tools to deal with the specific error sources discussed above can certainly make a more efficient use of the allocated computational resources.

Figure 4.26: Manual vs. automatic mesh determination in *Flowerpot*. *Top:*Planes recovered by the algorithm. *Bottom*: Planes recovered by manual selection of the points.

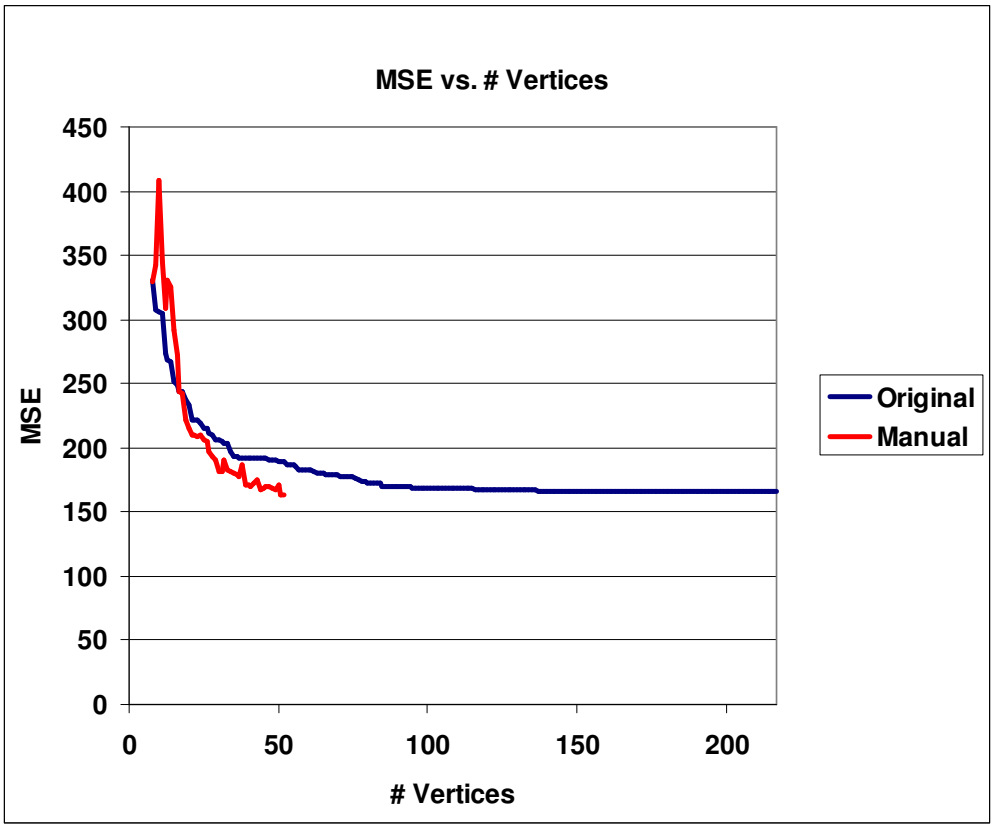


Figure 4.27: Manual vs. automatically generated mesh for *Flowerpot*. Sequential phase. Blue indicates the result for the proposed algorithm, red, for manually selected points.

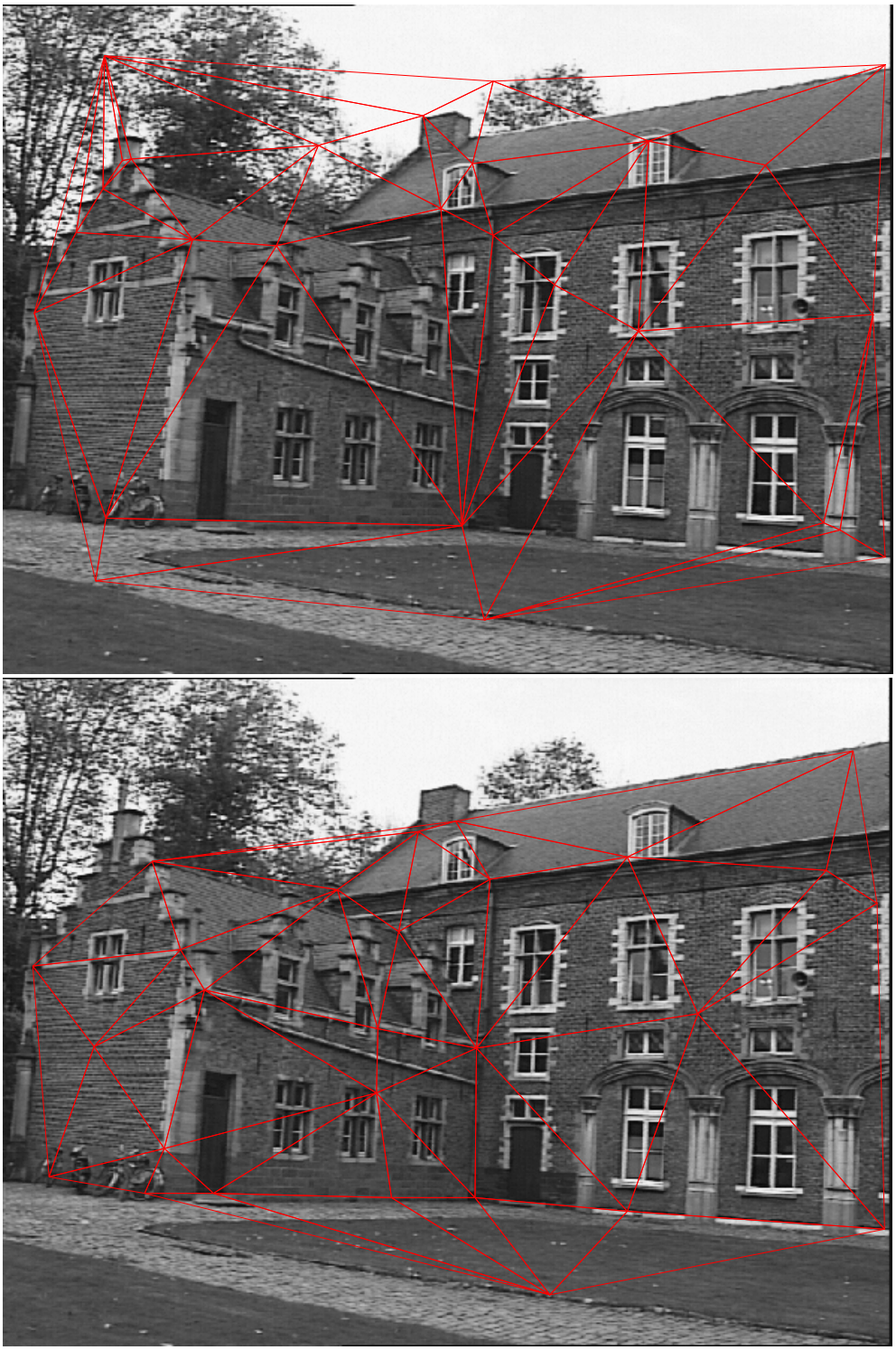


Figure 4.28: Manual vs. automatic mesh determination in *Castle*. *Top:*Planes recovered by the algorithm. *Bottom*: Planes recovered by manual selection of the points.

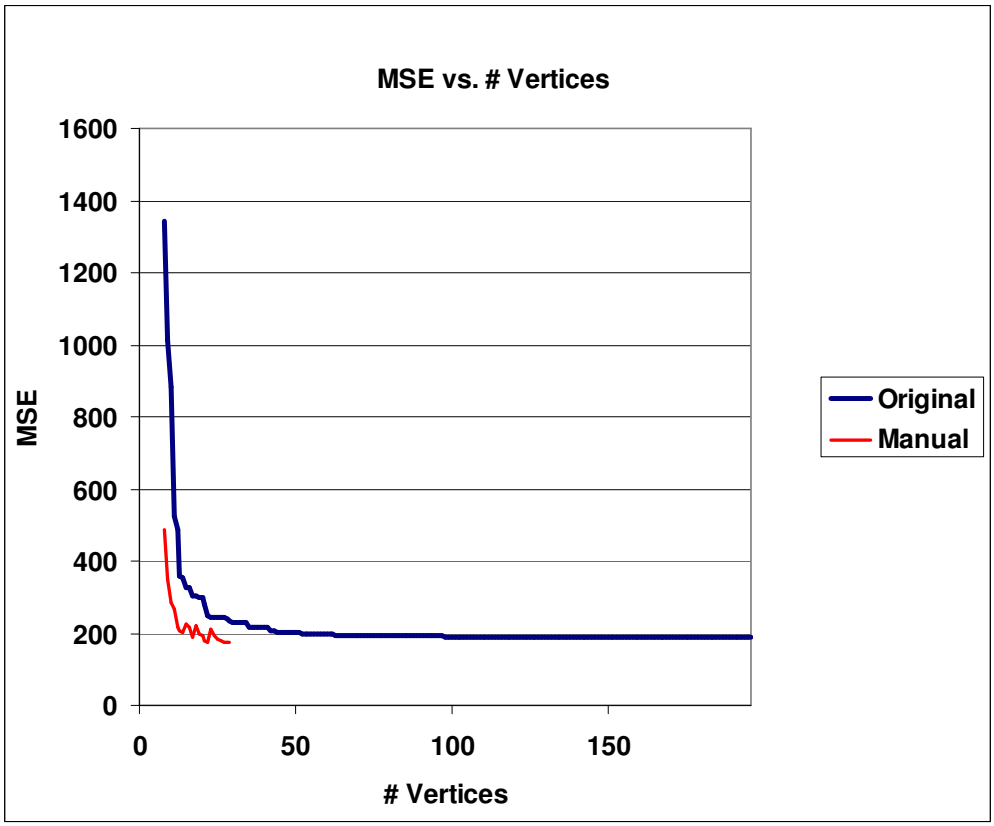


Figure 4.29: Manual vs. automatically generated mesh for *Castle*. Sequential phase. Blue indicates the result for the proposed algorithm, red, for manually selected points.

4.5.2 Stability of the Convergence Point

The experiments in the previous section clearly indicate that the proposed algorithm shows a convergent behavior. In an attempt to verify whether this behavior is consistent, and the point of convergence is stable, the sequential phase was disrupted by randomly rejecting a third of the features extracted by the algorithm. Moreover, these features were excluded from the algorithm for the rest of the sequential phase. Such a procedure yields a different vertex set, and consequently, a different mesh. For *Cliff, Breakdancers* and *Castle*, the above procedure was repeated multiple times. The median curves for the sequential phase, along with that of the original algorithm, are presented in Figure 4.30, 4.31 and 4.32. The results indicate that, especially in *Cliff* and *Castle*, both curves almost coincide with that of the unperturbed operation. This implies that different paths still lead to the same solution, and the resulting solution is stable. The small performance gap present in *Breakdancers*, when compared to *Cliff* and *Castle* stems from the fact that *Breakdancers* is not a texture-rich data, thus, unlike *Cliff* and *Castle*, the rejected feature pair cannot be replaced with an equally reliable one, causing a degraded performance.

4.5.3 Coarse-to-Fine vs. Fine-to-Coarse

In Section 4.1, merits of coarse-to-fine reconstruction over fine-to-coarse were discussed. As a supplementary to that discussion, both approaches were experimented on *Venus* and *Cliff*. The fine-to-coarse algorithm used in the experiment is, in a way, inverse of the coarse-to-fine algorithm of the proposed method. It uses the vertex set obtained by the coarse-to-fine algorithm. At each step, the algorithm finds the triangular patch with the best representation quality. Among the vertices of that triangle, the one which has the most conformity to the

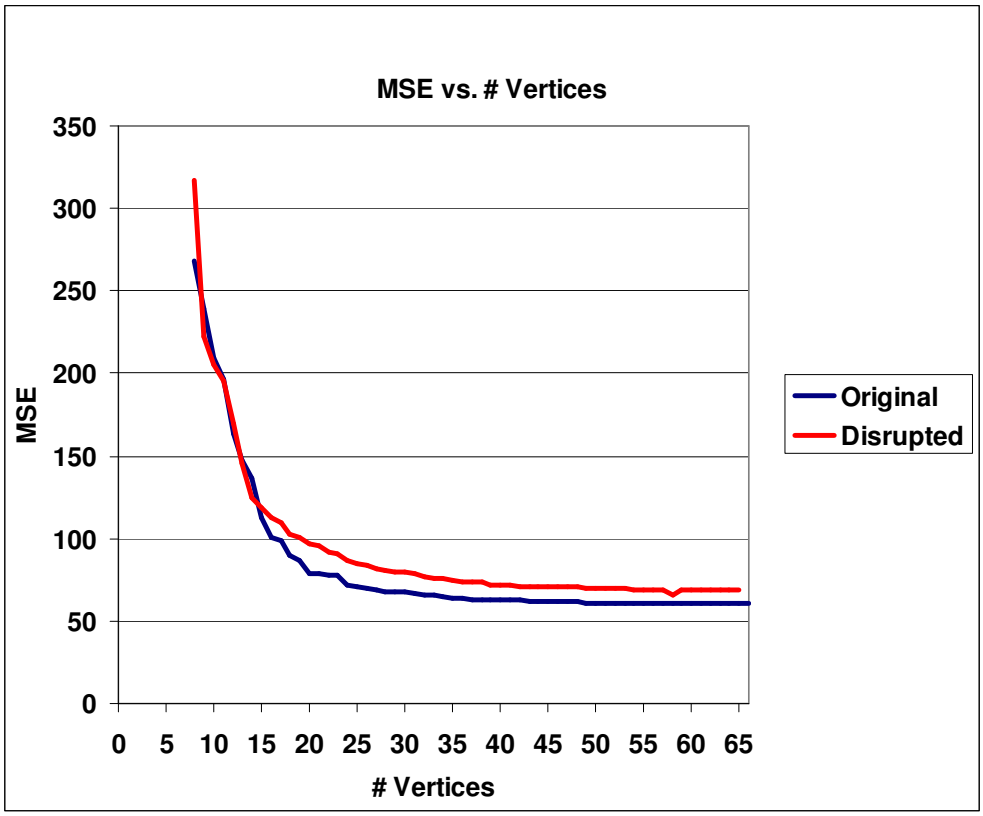


Figure 4.30: Stability test for *Breakdancers*. Blue line indicates the original process, and red, the average disrupted process.

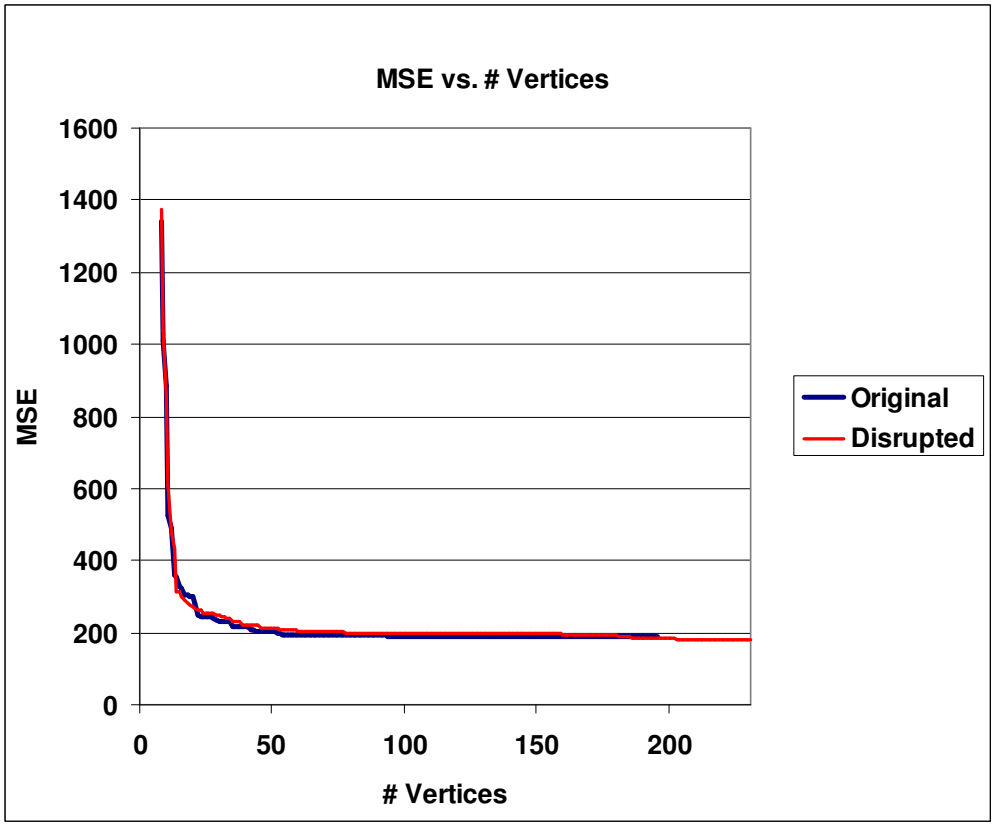


Figure 4.31: Stability test for *Castle*. Blue line indicates the original process, and red, the average disrupted process.

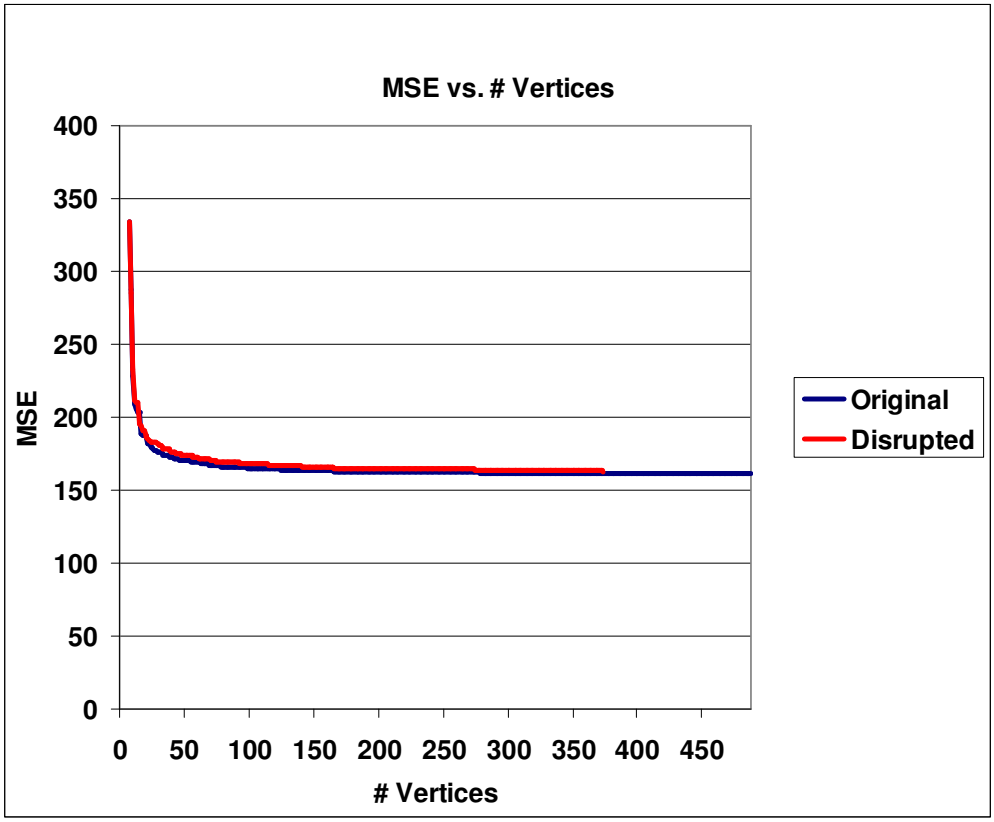


Figure 4.32: Stability test for *Cliff*. Blue line indicates the original process, and red, the average disrupted process.

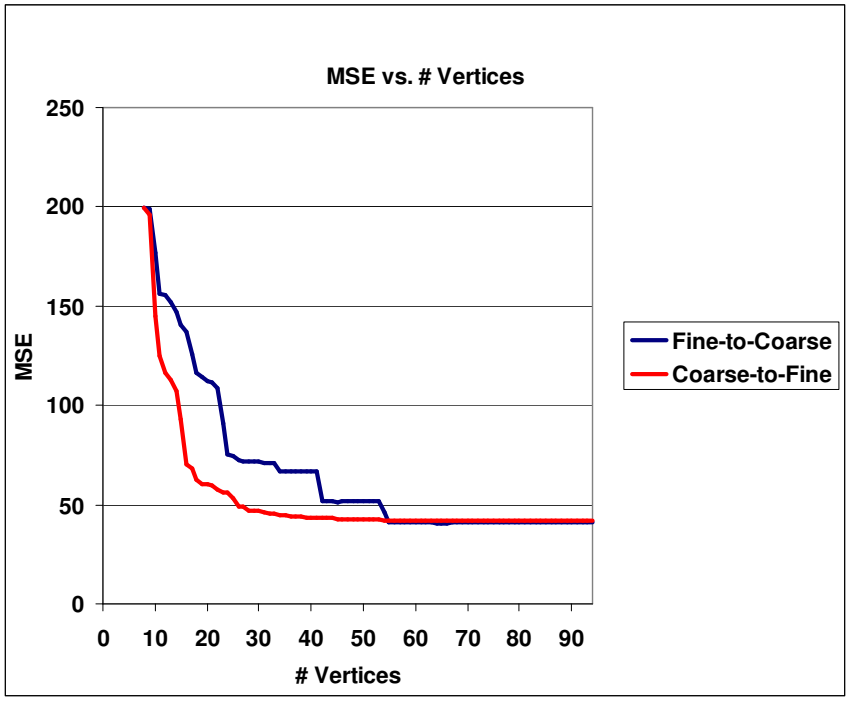
mesh synthesized in its absence, is discarded. This practice is equivalent to finding the most redundant vertex, among the vertices of the triangular patch. The experiment results are presented in Figure 4.33.

The experiment results are in support of the decision to employ a coarse-to-fine scheme. Fine-to-coarse scheme has a similar performance at high vertex numbers, however, for both data, when the number of vertices approach to 50, a performance gap favoring of coarse-to-fine scheme develops. This result is anticipated, when the rate-distortion efficient representation problem is interpreted as finding the smallest vertex set that describes the scene at a certain distortion. In this case, fine-to-coarse approach has a more complex error surface with more local minima, due to increased number of vertices, leading to poorer performance.

4.5.4 Vertex Selection: Geometry vs. Image Error

As discussed in Section 4.4, in order to pick the vertex that will provide the best improvement to the representation, symmetric transfer error (STE) is used. STE is a geometric distortion measure and evaluates the conformity of a vertex to the planar surface assumption. A high STE implies a vertex not in agreement with the mesh, therefore whose addition is likely to decrease the representation error measured by total square error (TSE). An alternative scheme involves directly picking the vertex that provides the largest decrease in TSE, among all vertices in a patch. This image-error based vertex selection scheme was implemented and tested on *Venus* and *Cliff*. The results are presented in Figure 4.34.

The experiment results indicate that, as expected, TSE yields better results. However, the difference is extremely small. Moreover, this marginal improvement is achieved at an extremely high computational cost. In order to have an idea about the order of magnitude of this cost increase, one should remember that STE



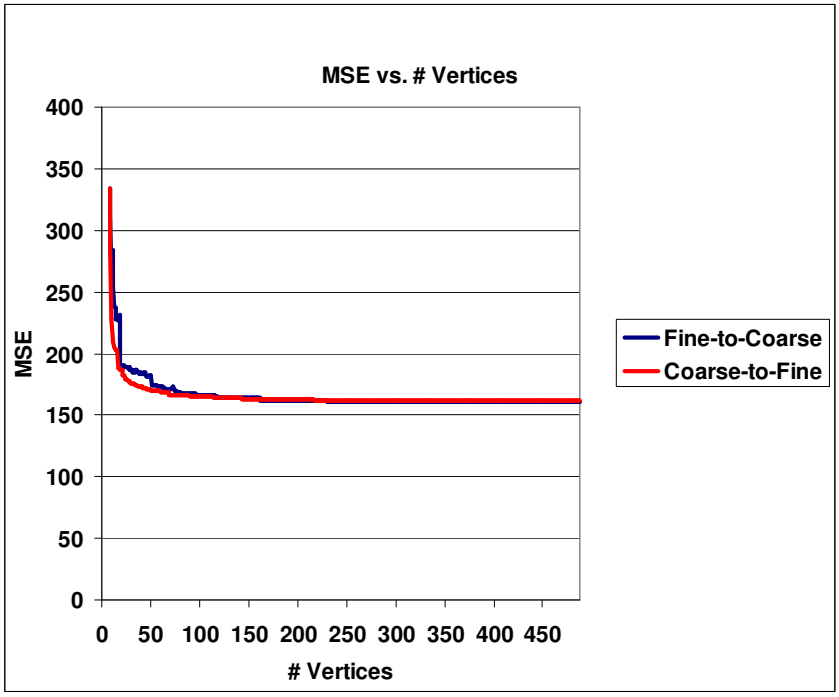


Figure4.33: Coarse-to-fine vs. Fine-to-coarse. *Top:* Progress of the sequential phase in *Venus*. *Bottom:* Progress of the sequential phase in *Cliff*.

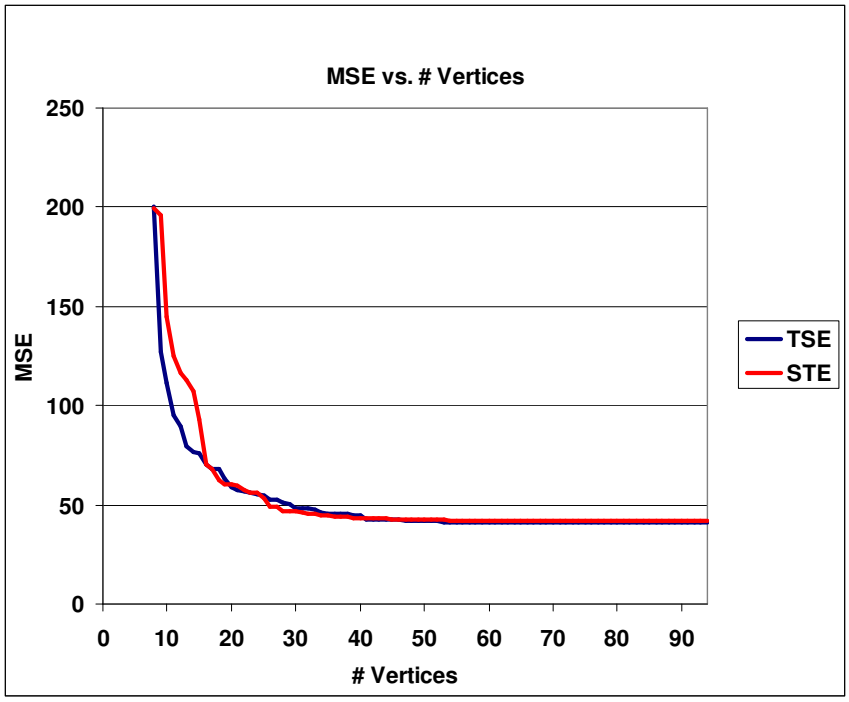
scheme performs two transfers per each vertex. However, the evaluation of TSE requires an update to the mesh and rendering the modified parts of the mesh. Rendering each pixel involves a transfer, plus bilinear interpolation. Therefore, in a typical image, the ratio of the computational complexities of TSE and STE schemes is in the order of thousands.

Another reason that justifies the use of STE to govern the vertex selection is mentioned the discussion on error metrics in Section 4.1. Minimizing the image error without any constraints on the geometry introduces a distortion to the structure estimate. STE acts to limit this distortion, by choosing vertices with respect to a geometric distortion criterion.

4.5.5 Rate-Distortion Performance

In a final experiment, the efficiency of the representation produced by the proposed method was compared with that of two other representations, dense depth map and *block motion vectors* (BMV). Dense depth map based representations describe a scene by the depth values of each pixel seen from a reference camera. Block motion vector based representations tile the reference frame into blocks, and assign a 2D motion vector for each block. The relevance of BMV-based representation to 3D scene representation problem stems from the fact that the 2D motion vector defines a mapping between the block in the reference frame and its correspondence in the target frame. This mapping can be expressed as

$$\mathbf{x}' \approx \begin{bmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}, \tag{4.10}$$



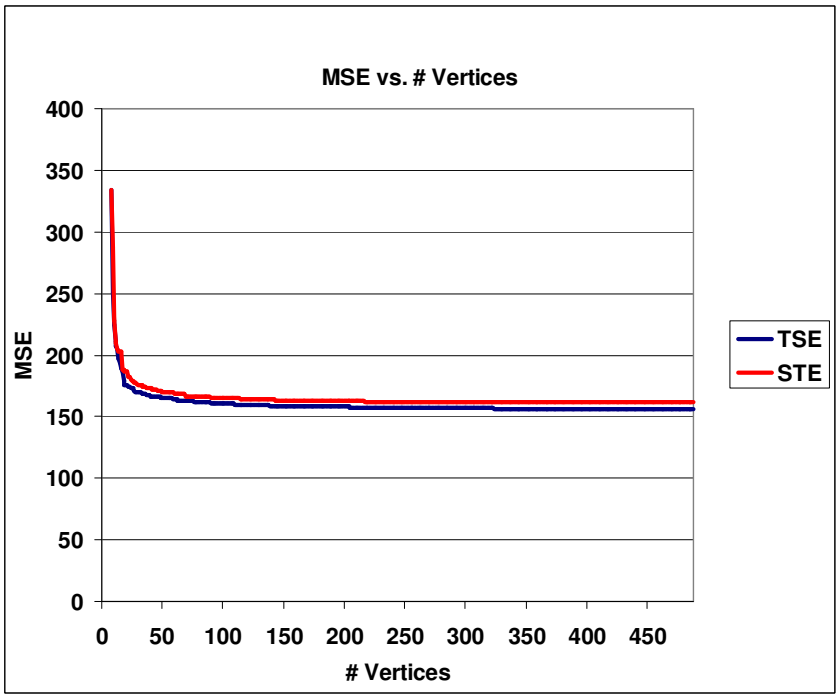


Figure 4.34: Vertex selection, STE vs. TSE. *Top:* Progress of the sequential phase in *Venus*. *Bottom:* Progress of the sequential phase in *Cliff*.

where **x** is a pixel in the reference block and **x'**, in the target block. *u* and *v* stand for the block motion vector. The mapping relating the blocks in both images is actually a homography, and as per the discussion in Section 4.3, this implies that the part of the 3D scene projecting to the block is approximated as a plane. Therefore, BMV-based representation models the 3D scene as a collection of planes. Apart from this fact, BMV-based prediction plays an important role in stereo image and video compression, thus comparing its performance to that of the proposed algorithm serves to assess the suitability of piecewise planar scene models for such applications in rate-distortion context.

The rate of the proposed algorithm was obtained by compressing the resulting mesh with *topological surgery* [91], a mesh encoder employed in ISO MPEG-4 standard [94]. The vertices were compressed with 20 bits. The distortion is measured by MSE. For a fair distortion comparison, only the regions that could be represented by all three methods were included during the error calculations.

In order to generate a dense depth map, the algorithm described in [81] was employed. The algorithm utilizes plane- and angle-sweeping to estimate a planar representation that minimizes the error between a target image and its prediction from a reference, while creating a dense depth map as a byproduct. The rate-distortion curve was prepared by compressing the resulting depth map, which was stored as a bitmap image, by ITU H.264 encoder [89][90], for different compression levels. The decompressed depth map was used to construct the target image from the reference. The rate-distortion curves for *Cliff, Venus, Breakdancers* and *Castle* are presented in Figures 4.35, 4.36, 4.37 and 4.38.

Block motion vectors were computed by ITU H.264 encoder, to make use of its advanced block motion vector estimation and compression engine. The encoder was configured to predict a target frame from a reference frame, i.e., to encode only two frames. In order to force the use of BMV, intra-frame mode was suppressed. This was achieved by setting its quantization parameter to 50, so that inter-frame prediction, i.e., block motion vectors, provides a better error for most

of the frame at the inter-intra selection step. The algorithm employs variable block sizes depending on the local characteristics of the frame and quarter-pixel resolution motion vectors. Motion vectors are encoded in a lossless fashion by using content-adaptive binary arithmetic coding (CABAC). During the experiments, the operational value of the rate was used loosely as the bit budget. The results of the experiment are presented in Table 4.8.

The dense depth map experiments clearly indicate the superiority of the proposed mesh-based method. Relatively high-bit-rate can be attributed to the fact that, although the depth images are remarkably smooth, therefore can achieve a low bit-rate, apparently it is a favorable trade-off to represent the structure with a small number of high-precision vertices, instead of many low precision transform coefficients. As for the distortion, there are two mechanisms in play: Compression artifacts and quantization losses. The former smooths depth discontinuities, causing distortions that contribute significantly to the final prediction error, but concentrated at the boundary (therefore, perceptually harder to notice, lending support to the arguments about the inadequacy of PSNR). The latter arises from the fact that a pixel can only have a value that comes from a discrete set of intensity levels, therefore the continuous depth values must be quantized, introducing an error into the depth values. This effect becomes stronger as the depth range increases. In the experiments, the depth range was quantized uniformly.

BMV experiments, on the other hand, present a more complex picture, which makes sense once the strengths and weaknesses of the BMV representation are considered. BMV provides a 2D scene representation, as it utilizes 2D motion vectors to describe the scene. Therefore, the descriptive power of the BMV representation degrades in cases where the effect of depth is non-negligible, such as scenes with a large depth range. On the other hand, the piecewise planar representation proposed in this work uses meshes, which is a 3D scene representation, therefore can successfully handle such scenes. Another effect of

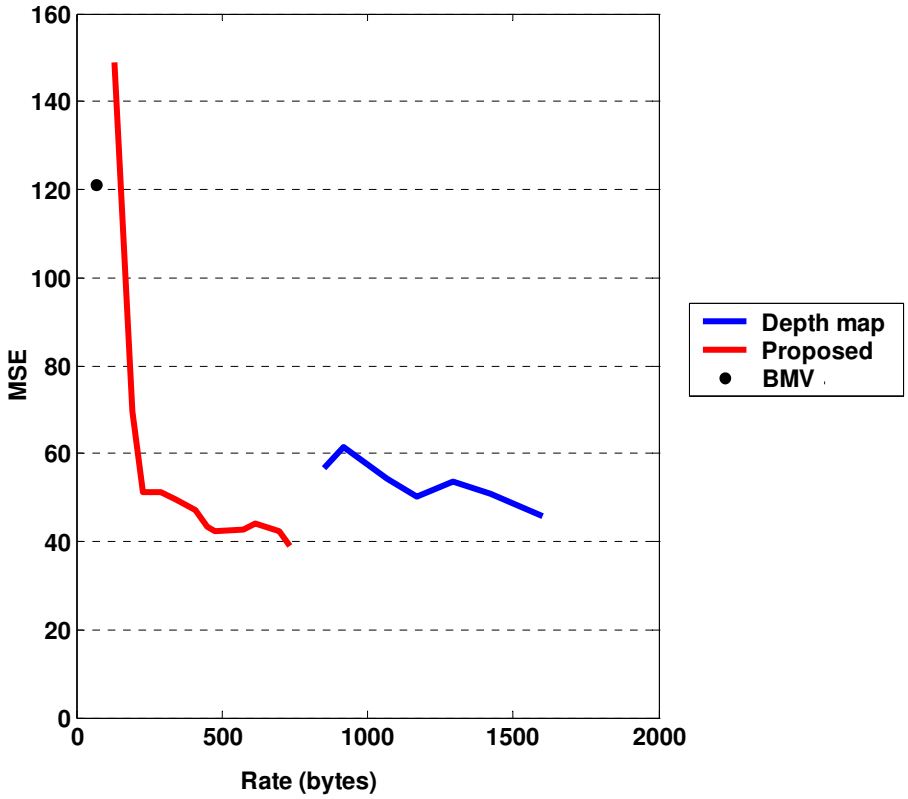


Figure 4.35: Rate-distortion plot for *Breakdancers*. Proposed method, depth map and BMV.

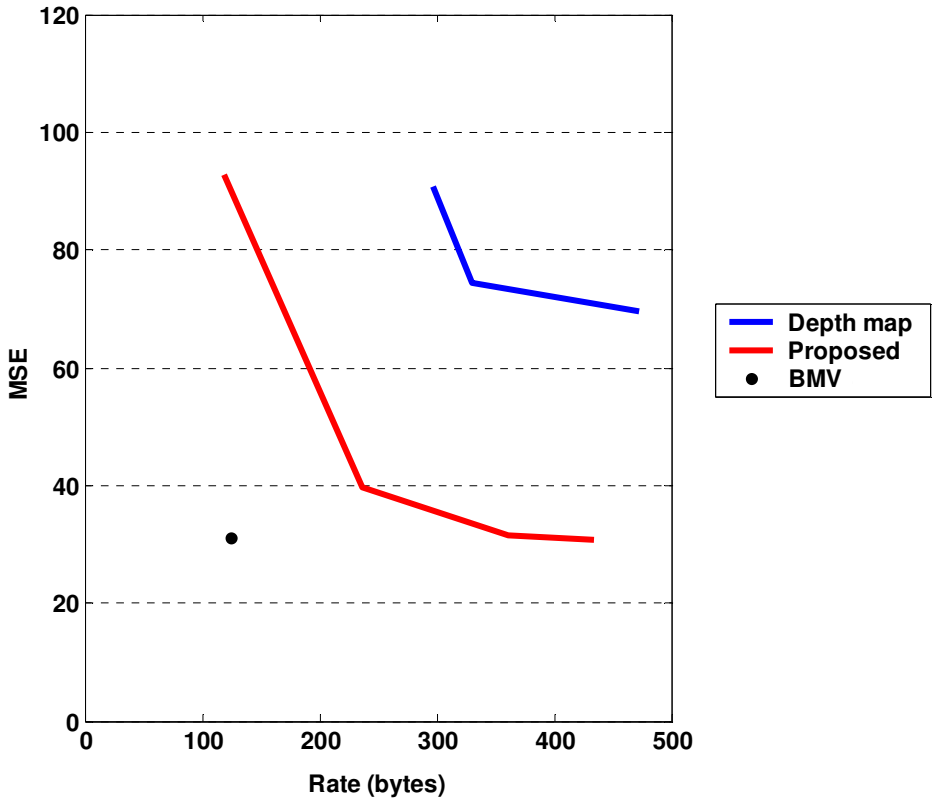


Figure 4.36: Rate-distortion plot for *Venus*. Proposed method, depth map and BMV.

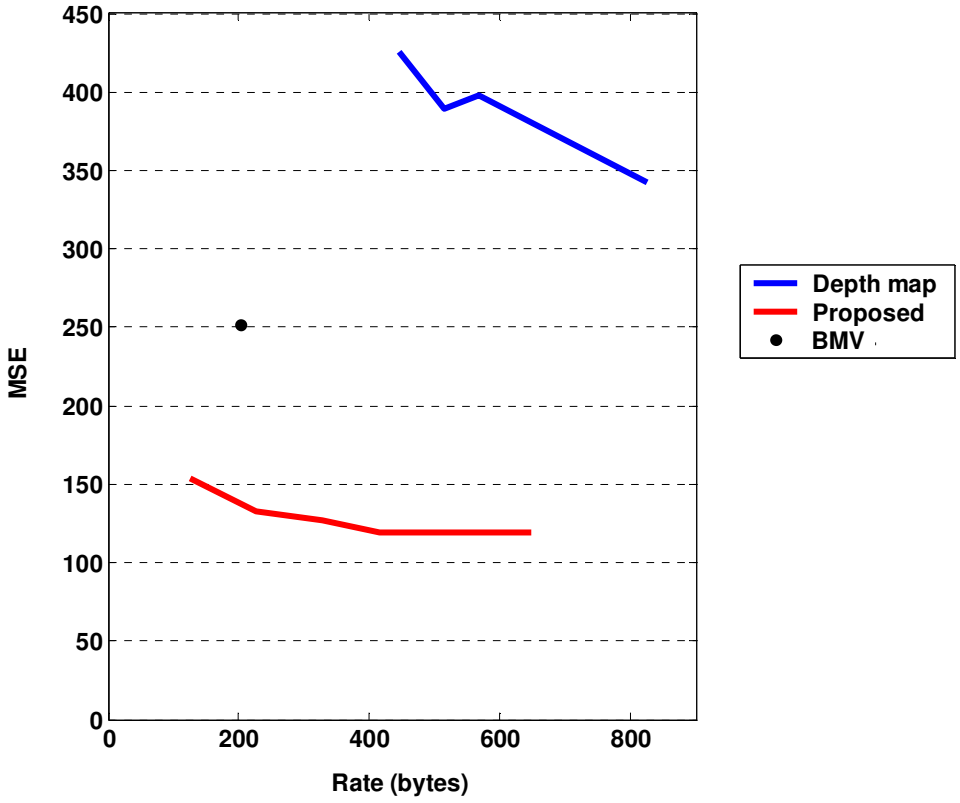


Figure 4.37: Rate-distortion plot for *Cliff*. Proposed method, depth map and BMV.

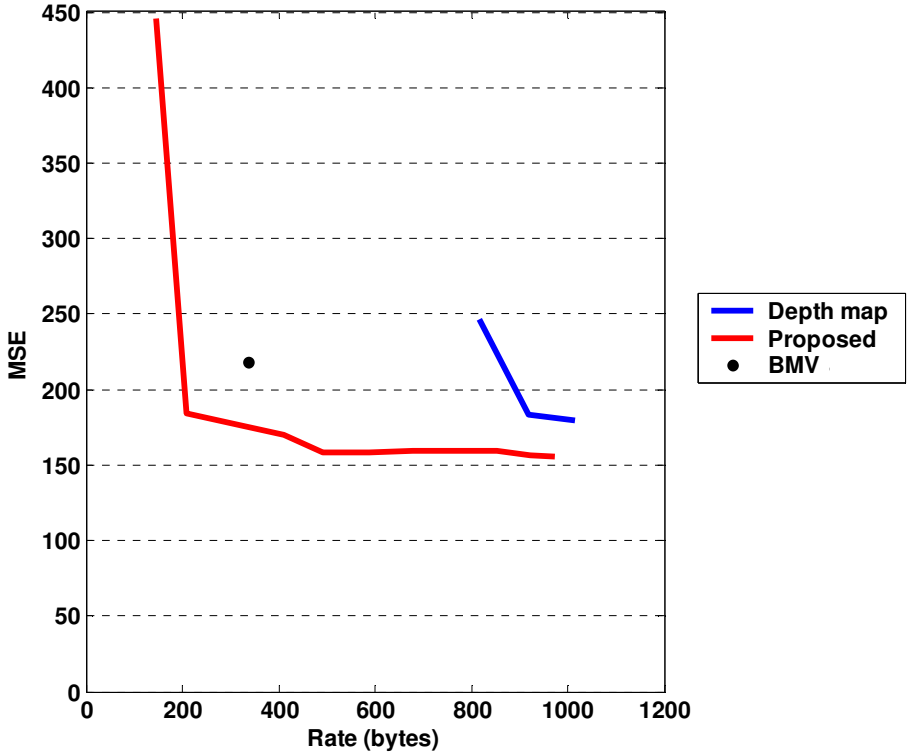


Figure 4.38: Rate-distortion plot for *Castle*. Proposed method depth map and BMV.

Table 4.8: Rate-distortion performances of block motion vectors and proposed method

	Block Motion Vectors		Proposed	
	Rate (Bytes)	**Distortion (MSE/PSNR)** (Equation 2.1)	**Rate (Bytes)**	**Distortion (MSE/PSNR)** (Equation 2.1)
Wall	210	408.78/22.01	238	**159.23/26.02**
Wadham	514	**195.31/25.22**	500	316.21/23.13
Castle	337	217.83/24.75	329	**175.84/25.14**
Flowerpot	217	211.94/24.86	211	**206.79/24.98**
Cliff	204	251.56/23.95	227	**132.21/26.92**
Breakdancers	69	**120.66/27.31**	133	148.78/26.41
Venus	125	**31.55/33.14**	119	92.90/28.45
Palace	317	265.44/23.89	330	**156.57/26.18**

depth range becomes evident from its interpretation as disparity range. In block motion vector computation, a large disparity range requires a large search region, and this increases the possibility of incorrect matches and block motion vectors, causing distortions in the reconstruction. Moreover, a large depth range suggests a larger variation of depth, hence disparity values within a block, which is a violation of the uniform disparity assumption. These conclusions are supported by a comparison of the performance of the BMV and mesh-based representations in large depth-range data, such as *Cliff, Palace* and *Wall,* and small-depth range data, e.g., *Venus* and *Breakdancers*. The proposed algorithm outperforms the BMV representation in the former case, and is inferior to it in the latter, especially in *Venus*.

Another issue that should be considered is the case of scenes with disconnected planes. The BMV representation successfully handles such scenes, as each block is registered independently from its neighbors; therefore, the scene is modeled as a collection of disjoint planes. On the other hand, the proposed algorithm does not accommodate for such cases, which is another factor that contributes to its inferior

performance to the BMV representation in *Venus*, a scene composed of disjoint planes.

The reasons for the failure of the proposed algorithm in *Wadham* were discussed in Section 4.5.1.

4.6 Conclusion

This chapter focuses on the scene representation module of the 2D-3D uncalibrated video conversion system proposed in this dissertation. To this aim, first the niche of efficient scene representation and the lack of emphasis on the efficiency in the literature are discussed. Then, the design guidelines for a rate-distortion efficient piecewise-planar scene reconstruction algorithm are laid out. Following the description of the two main blocks, dynamic mesh construction and view rendering from a piecewise planar representation, a novel piecewise scene reconstruction algorithm is proposed. The algorithm is capable of building a representation from two views of a scene. The representation is gradually constructed by adding vertices to a mesh in such a way that, the intensity error between the target image and its prediction is minimized. Since this approach relates the quality and the complexity of the representation, the algorithm is a rate-distortion efficient procedure. The result is further refined by the application of steepest descent or simulated annealing. The proposed algorithm is demonstrated to be capable of achieving representations of scenes with as low as 30-50 vertices in some cases, and to be superior to dense depth map representation. As for BMV representation, the algorithm provides a viable alternative to the cases in which the assumptions of the BMV representation do not hold, which is not an uncommon occurrence for uncontrolled data acquisition cases, such as broadcast 2D video.

CHAPTER 5

SUMMARY AND CONCLUSION

This dissertation presents an almost complete system for 2D to 3D conversion of uncalibrated video, requiring only a 2D video sequence, pre-segmented into shots, from a moving camera as an input, and producing a sparse 3D point cloud, a dense piecewise planar 3D representation and the trajectory of the camera as its output. The mesh-based representation can be used to render views of the imaged scene, and is rate-distortion efficient, facilitating its use in multi-view coding applications. The issue that makes the proposed solution "almost complete" is the fact that the dynamic scenes case is not fully addressed. The conversion chain is composed of several distinct modules, dedicated to specific tasks, namely, feature extraction and tracking, sparse structure and camera matrix estimation, and dense scene representation. Below is a brief summary of the research and the conclusions.

5.1 Summary and Conclusions

The feature extraction and tracking module serves to track features extracted from a video sequence by Harris corner detector, to establish long and accurate trajectories for reliable sparse structure and camera matrix estimation. In order to achieve this goal, two algorithms, the CC and OF trackers, are proposed. The

algorithms are specifically designed to exploit the properties of video inputs. While the algorithms utilize tools well-known in computer vision literature, such as KLT and Kalman filter, the CC tracker also serves to introduce *auction* technique from radar tracking literature to image feature tracking problem.

The performances of both trackers are studied by a set of heuristics reflecting various aspects of the trajectory set. The major conclusions revealed by the experiments are as follows:

- The use of Kalman filter considerably improves the intensity pattern tracking performance.
- The CC tracker recovers a more reliable trajectory set when tracking intensity patterns.
- The OF tracker produces a more populous and longer trajectory set.
- In the estimation of F-matrix, when the trajectory accuracy is above a certain level, the number of correspondences and the baseline length determine the performance of the estimator. This implies that, the OF tracker is better suited for 3D reconstruction task.

The sparse structure and camera module aims to solve the MFSfM problem to obtain the initial structure and camera matrix estimates for dense reconstruction. The research on this topic reveals that while there exist well-established solutions for sequential reconstruction for a given sequence, the problem of determining the sequence itself is largely an unsolved problem, especially in video case, where the inherent (temporal) ordering of the sequence is not suitable for reconstruction. In order to solve this problem, a novel metric is designed. The metric assesses all frame pairs in a sequence with respect to the amount of structure seen in the pair and amenability to 3D reconstruction, via a nonlinear function of number of correspondences and baseline length. The proposed metric is employed in a

generalization of the conventional sequential reconstruction procedure, i.e., *prioritized sequential reconstruction*. *Prioritized sequential reconstruction* extends the conventional algorithm, which operates by locating the camera corresponding to a new frame with respect to a reference reconstruction, and updating the structure with the points seen by the camera, to the case in which multiple subsequences are processed in parallel. This is achieved by an algorithm that supervises the initiation, update and merger of multiple sequential reconstructions to attempt a more effective exploration of the solution space.

The sparse reconstruction problem led to side projects, such as self-calibration and segmentation of feature trajectories for the reconstruction of dynamic scenes. However, metric reconstruction without accurate internal calibration matrices was observed to degrade the estimate quality substantially, without any benefits other than visualization. Hence, the study on self-calibration is not pursued further than an initial exploration. The dynamic scene reconstruction problem was the focus of a collaborative research effort, and it was concluded when segmentation and sparse reconstruction of each element was demonstrated to be possible in [85]. The focus of the work presented in this dissertation is static scenes, and support for dynamic scenes is lacking in the dense reconstruction module.

The work and experiments on sparse reconstruction leads to the following conclusions:

- Any sequential algorithm attempting to solve the sparse reconstruction problem should be operated at projective ambiguity level, to remove the effect of the internal calibration errors on the reconstruction process. The upgrade to metric reconstruction should be attempted after a reliable projective reconstruction and camera set is achieved.

- The frame pair prioritization by a weighted sum of baseline distance and number of correspondences improves the performance of the sequential 3D reconstruction stage [82].
- The nonlinear weighting scheme is superior to its immediate alternatives, linear weighting scheme and baseline-only ordering [82].
- The simultaneous exploration of the solution space by multiple reconstructions, as opposed to a single reconstruction, makes the search procedure more robust to unsatisfactory local minima.

Dense reconstruction problem is studied in the context of rate-distortion efficiency, since it was observed that in the literature, the efficiency aspect of dense scene representations has been underemphasized. To this aim, a novel rate-distortion efficient piecewise planar reconstruction algorithm, which gradually builds a representation by adding vertices to a mesh in a way to minimize the difference between a target image and its prediction, is proposed [83]. The proposed algorithm offers an alternative to the exclusive use of image error, which is identified to be a problem that can severely distort the structure estimate, by injecting a geometric component through the use of symmetric transfer error for vertex selection. Another novel approach introduced in this work is the identification of not only vertices, but also the cameras as a source of error, and inclusion of the camera parameters in the parameterization for the non-linear optimization stage.

The extensive experiments conducted on various data sets using the proposed algorithm indicate that:

- The proposed method successfully produces a rate-distortion efficient scene representation. In rate-distortion sense, the mesh-based representation generated by the algorithm is superior to dense depth map

based representation, and in large depth-range cases, to block motion vectors.

- The assumption that the target image can be predicted from the reference image can become invalid, locally because of introduction of new intensity patterns due to changing viewpoint, and globally because of lighting changes. This significantly degrades the performance of the algorithm.
- Meshing errors due to the construction of the mesh on a 2D-plane, insufficiently sampled boundary regions, and the violation of the connectedness assumption introduce effects such as bending, and give rise to visually disturbing artifacts.
- Symmetric transfer error successfully predicts the vertices that should be added to the representation to reduce the image prediction error.
- Coarse-to-fine approach is superior to fine-to-coarse approach for building rate-distortion efficient scene representations.

5.2 Future Directions

The main contributions of this work, i.e., prioritized sequential 3D reconstruction and rate-distortion efficient piecewise planar scene reconstruction, were inspired by the challenges posed by the various aspects of the 2D-to-3D conversion problem. While this dissertation describes a powerful framework for the solution of this problem, many issues remain, whose resolution will substantially improve the performance and robustness of the process. Below is a, by no means complete, collection of pointers for future research efforts:

Statistics of matching problem: The propagation of the variance of feature position through the reconstruction chain up to the corresponding 3D point is a well-studied topic [10]. However, accurate estimation of the bias, variance and the probability distribution of the uncertainty of the feature position estimates, beyond

the ubiquitous zero-mean Gaussian assumption remains a challenge. Moreover, the statistical characterization of the matching process, an important topic for the evaluation of the reliability the corresponding pairs is not yet studied in computer vision discipline, except for the attempts to relate the match quality measures empirically to prior probabilities [38][86].

Improved key-point detection: While Harris corner detector is employed for feature extraction in this work, the popularity of SIFT among researchers is on the rise. However, in a video problem, SIFT has three drawbacks: The computational complexity, the imposition of unnecessary invariance conditions and the order-dependency of the matching. Therefore, an improved SIFT that can deal with these issues to obtain better features is certainly in demand.

Learning RANSAC: RANSAC, and generally all *X-SAC* algorithms underutilize the information recovered by rejected hypotheses. A learning RANSAC scheme can keep this information by possibly penalizing the data participating to the rejected hypotheses, thus reducing their probability of being selected in further attempts, and weighting the more reliable data.

Improved prioritization metric: The prioritization metric proposed in this work can be more closely related to covariance of the structure and camera matrix estimates. Moreover, a measure that does not need an initial metric reconstruction is desirable to remove the dependency on calibration information completely.

Point-dependent reconstruction order: The accuracy of a structure point estimate depends on the distance between the camera centers, and the angle between the rays intersecting at the point. This implies that no frame pair is the best for all structure points seen in the frames. Therefore, the prioritization idea

can be pursued further to build frame pair sequences for individual structure points.

Use of image analysis in piecewise planar reconstruction: Image analysis can provide edge and segmentation information. This information can be incorporated into the dense reconstruction process to determine the dominant planes and strong edges in the scene, and to provide a better initial mesh. The viability of such approaches was shown in [87] and [88].

Depth map aided piecewise planar reconstruction: The proposed piecewise planar reconstruction algorithm assumes a continuous scene with sufficient features. The regions without features cannot be represented, and the discontinuities cannot be modeled. These issues can be addressed by coupling the algorithm with a depth-map based planar reconstruction scheme. One technique capable of dealing with such issues is [81], which can represent the scene as a collection of disconnected planes, and incorporate this information with motion and color to achieve segmentation. The proposed algorithm can be used to represent the segments, recover the finer structures on them, and to form a collection of disconnected surfaces.

Performance measures for dense reconstruction: The performance of dense reconstruction algorithms are often measured by PSNR, an image error metric. However, as mentioned in Chapter 4, this metric disregards the structure, and minimization with respect to PSNR actually deforms the structure in the presence of erroneous camera parameters. Therefore, there is a need for performance measures that represent both the image and the structure deformations, for structure recovery problems.

REFERENCES

[1] D. Nister, "Automatic Passive Recovery of 3D from Images and Video", *Proceedings of 2nd International Symposium on 3D Data Processing, Visualisation and Transmission,* Vol.00, pp. 438-445, 2004.

[2] M. Pollefeys, "Tutorial on 3D Modeling from Images", *6th European Conference on Computer Vision*, Tutorial Notes, 2000.

[3] T. Rodriguez, P. Sturm, P. Gargallo, N. Guilbert, A. Heyden, F. Jauregizar, J. M. Menendez, J. I. Ronda, "Photorealistic 3D Reconstruction from Hand-Held Cameras", *Machine Vision and Applications*, Vol. 16, No. 4, pp. 246-257, 2005.

[4] İ. Bayram, *Interest Point Matching Across Arbitrary Views*, M.S. Thesis, Middle East Technical University, Turkey, 2004.

[5] J. Repko, M. Pollefeys, "3D Models from Extended Uncalibrated Video Sequences: Addressing Key-Frame Selection and Projective Drift", *Proceedings of 5th International Conference on 3D Digital Imaging and Modeling*, pp. 150-157, 2005.

[6] P.H S. Torr, "Geometric Motion Segmentation and Model Selection", *Philosophical Transactions of Royal Society of London A,* Vol. 356, pp. 1321–1340, 1998.

[7] C.G. Harris and M. Stephens, “A Combined Corner and Edge Detector”, *Proceedings of 4th Alvey Vision Conference*, pp. 147-151, 1988.

[8] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints”, *International Journal of Computer Vision*, Vol. 60, No. 2, pp.91-110, 2004.

[9] J.-Y. Bouguet, “Pyramidal Implementation of the Lucas Kanade Feature Tracker”, Intel Corporation, Microprocessor Research Labs OpenCV Documents, 2000.

[10] R. Hartley, A. Zisserman, *Multiple View Geometry*, Cambridge University Press, UK, 2003.

[11] M. Fischler and R. Bolles, “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”, *Communications of the ACM*, Vol. 24, No. 6, pp. 381-385, 1981.

[12] R. I. Hartley, “In Defence of the 8-Point Algorithm”, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 19, No. 6, pp. 580-593, 1997.

[13] L. Kitchen and A. Rosenfeld, “Gray Level Corner Detection”, *Pattern Recognition Letters I*, No.2, pp.95-102, 1982.

[14] Z. Zhang, R. Deriche O.D. Faugeras, Q.-T. Luong, "A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry", *Artifical Intelligence*, vol. 78, No. 1-2, pp. 87-119, 1995.

[15] J. Shi, C. Tomasi, “Good Features to Track”, *Proceedings of 1994 IEEE Conference on Computer Vision and Pattern Recongnition*, pp. 593-600, 1994.

[16] B.D. Lucas and T. Kanade, “An Iterative Image Registration Technique with an Application to Stereo Vision", *DARPA Image Understanding Workshop*, 1981, pp121-130.

[17] M. Irani, P. Anandan, “A Unified Approach to Moving Object Detection in 2D and 3D Scenes”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 6, pp. 577-589, 1998.

[18] G. Qian, R. Chellappa, Q. Zheng, “Bayesian Algorithms for Simultaneous Structure from Motion Estimation of Multiple Independently Moving Objects”, *IEEE Transactions on Image Processing*, Vol. 14, No.1, pp. 94-109, 2005.

[19] Y. Weiss, “Segmentation Using Eigenvectors: A Unifying View”, *Proceedings of 7th IEEE International Conference on Computer Vision*, pp. 975-982, 1999

[20] W. Fitzgibbon, A. Zisserman, “Multibody Structure and Motion: 3D Reconstruction of Independently Moving Objects”, *Proceedings of 6th European Conference on Computer Vision*, 2000.

[21] R. Vidal, S. Soatto, Y. Ma, S. Sastry, “Two-view Multibody Structure from Motion”, *Technical Report UCB/ERL M02/02*, 2002.

[22] O. Faugeras, Q. Luong, S. J. Maybank, "Camera Self-Calibration: Theory and Experiments", *Proceedings. of 2nd European Conference on Computer Vision*, pp. 321-334, 1992.

[23] Q.Luong, O. Faugeras, "Self-Calibration of a Moving Camera from Point Correspondences and Fundamental Matrices", *International Journal of Computer Vision,* Vol. 22, No. 3, pp. 261-289, 1997.

[24] M. Pollefeys, L. Van Gool, "A Stratified Approach to Self-Calibration", *Proceedings of 1997 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 407-412, 1997.

[25] P. Sturm, "On Focal Length Calibration from Two Views," *Proceedings of.2001 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 145-150, 2001.

[26] R. Hartley, "Estimation of Relative Camera Positions for Uncalibrated Cameras", *Proceedings of 2nd European Conf. on Computer Vision*, pp. 579-587, 1992

[27] P. R. S. Mendonca and R. Cipolla, "A Simple Technique for Self-Calibration", *Proceedings of 1999 IEEE Conference on Computer Vision and Pattern Recognition,* Vol. 1 , pp. 500-505, 1999.

[28] D. Nister, "Untwisting a Projective Reconstruction", *International Journal of Computer Vision,* Vol 60., No. 2, pp. 165-183, 2004.

[29] H. Li, "A Simple Solution to the Six-Point Two-View Focal Length Estimation Problem", *Proceedings of 12th European Conference on Computer Vision*, Vol. 4, pp. 200-213, 2006.

[30] J. Weng, T. S. Huang, N. Ahuja, "Motion and Structure from Two Perspective Views: Algorithms, Error Analysis and Error Estimation", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 11, No. 5, 1989.

[31] H. C. Longuet-Higgins, "A Computer Algorithm for Reconstructing a Scene from Two Projections", *Nature,* No. 293, pp. 133-135, 1981.

[32] P. H. S. Torr, A. Zisserman, "Robust Computation of Multiple View Relations", *Proceedings of IEEE 3rd International Conference on Computer Vision,* pp. 727-732, 1998.

[33] P. H. S. Torr, *Motion Segmentation and Outlier Detection,* PhD Thesis, University of Oxford, 1995.

[34] Z. Zhang, "Determining the Epipolar Geometry and its Uncertainty: A Review", *Technical Report 2927,* INRIA, France, 1996.

[35] G. Csurka, C. Zeller, Z. Zhang, O. D. Faugeras, "Characterizing the Uncertainty of the Fundamental Matrix", *Computer Vision and Image Understanding,* Vol. 68, No. 1, pp. 18-36, 1996.

[36] R. Hartley, P. Sturm, "Triangulation", *Computer Vision and Image Understanding,* Vol. 68, No. 2, pp. 146-157, 1997.

[37] D. Nister, "An Efficient Solution to Five-Point Relative Pose Problem", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 26, No. 6, pp. 756-770, 2004.

[38] O. Chum, J. Matas, "Matching with PROSAC: Progressive Sample Consensus", *Proceedings of 2005 IEEE Conference on Computer Vision and Pattern Recognition,* Vol. 1, pp. 220-226, 2005.

[39] O. Chum, T. Werner, J. Matas, "Two-View Geometry Estimation Unaffected by a Dominant Plane", *Proceedings of 2005 IEEE Conference on Computer Vision and Pattern Recognition,* Vol. 1, pp. 772-779, 2005.

[40] C. Tomasi, T. Kanade, "Shape and Motion from Image Streams: A Factorization Method", *Technical Report CS-92-104*, Carnegie Mellon University, USA, 1992.

[41] M. Han, T. Kanade, "Perspective Factorization Methods for Euclidean Reconstruction*", CMU-RI-TR-99-22*, Carnegie Mellon University, USA, 1999.

[42] J. P. Costeira, T. Kanade, "A Multibody Factorization Method for Independently Moving Objects", *International Journal of Computer Vision*, Vol. 29, No. 3, pp. 159-179, 1998.

[43] J. Yan, M. Pollefeys, "A Factorization Approach to Articulated Motion Recovery", *Proceedings of 2005 IEEE Conference on Computer Vision and Pattern Recognition,* Vol. 2, pp. 815-821, 2005.

[44] H. Jia, J. Fortuna, A. M. Martinez, "Perturbation Estimation of the Subspaces for Structure from Motion with Noisy and Missing Data", *Proceedings of 4th International Symposium on 3D Data Processing, Visualisation and Transmission,* pp. 1101-1107, 2006.

[45] Z. Sun, V. Ramesh, A. M. Tekalp, "Error Characterization of the Factorization Method", *Computer Vision and Image Understanding,* Vol. 82, No. 2, pp. 110-137, 2001.

[46] M.I.A. Lourakis , A.A. Argyros , "The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm", *Technical Report 340*, FORTH-ICS, Greece, 2004.

[47] M.I.A. Lourakis , A.A. Argyros , "Is Levenberg-Marquardt the Most Efficient Optimization Algorithm for Implementing Bundle Adjustment",*Proceedings of 10th International Conference on Computer Vision,* vol. 2, pp.1526-1531, 2005.

[48] J. Zhang, D. G. Aliaga, M. Boutin, R. Insley, "Angle Independent Bundle Adjustment Refinement", *Proceedings of 4th International Symposium on 3D Data Processing, Visualisation and Transmission*, pp. 1108-1116, 2006.

[49] B. Triggs, P. F. McLauchlan, R. I. Hartley, A. W. Fitzgibbon, "Bundle Adjustment- A Modern Synthesis", *Lecture Notes in Computer Science,* Vol. 1883, pp. 298-372, 1999.

[50] S. Soatto, R. Frezza, P. Perona, "Motion Estimation on the Essential Manifold", *Procedings of 3rd European. Computer Vision Conference*, pp. B:61–B:72, 1994.

[51] A. Azarbayejani, A. Pentland, "Recursive Estimation of Motion, Structure and Focal Length", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 6, pp. 562-575, 1995.

[52] T.J. Broida, R. Chellappa, "Experiments and Uniqueness Results on Object Structure ad Kinematics from a Sequence of Monocular Images", *Proceedings of Workshop on Visual Motion*, pp.. 21-30, 1989.

[53] A. D. Chowdury, R. Chellappa, "Statistical Bias in 3D Reconstruction from Monocular Video", *IEEE Transactions on Image Processing*, Vol. 14, No. 8, pp. 1057-1062, 2005.

[54] G. Qian, R. Chellappa, "Structure from Motion Using Sequential Monte Carlo Methods", *International Journal of Computer Vision*, Vol. 59, No. 1, pp. 5-31, 2004.

[55] S. Soatto, P. Perona, "Reducing 'Structure from Motion': a General Framework for Dynamic Vision Part 1: Modeling", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 20, No. 9, pp. 943-960, September 1998.

[56] P. Strum, B. Triggs, "A Factorization Based Algorithm for Multi-Image Projective Structure and Motion", *Proceedings of the 4th European Conference on computer Vision*, Vol. 2, pp. 709-720, 1996.

[57] J.I. Thomas, J. Oliensis, "Dealing with Noise in Multiframe Structure from Motion", *Computer Vision and Image Understanding*, Vol. 76, No. 2, pp. 109-124, 1999.

[58] A. Smolic, K. Mueller, N. Stefanoski, J. Ostermann, A. Gotchev, G. B. Akar, G. Triantafyllidis, A. Koz, "Coding Algorithms for 3DTV- A Survey*",* to appear in *IEEE Transactions on Circuit Systems and Video Technology Special Issue on MVC*.

[59] D. Scharstein, R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms", *International Journal of Computer Vision*, Vol. 47, No. 1-2-3, pp. 7-42, 2002.

[60] A. Bartoli, P. Sturm, and R. Horaud, "A Projective Framework for Structure and Motion Recovery from Two Views of a Piecewise Planar Scene", *Technical Report RR-4970,* INRIA, 2000.

[61] K. Schindler, "Spatial Subdivision for Piecewise Planar Object Reconstruction*", Proceedings. of SPIE and IS&T Electronic Imaging-Videometrics VIII*, pp. 194-201, 2003.

[62] R. Musin, "Properties of the Delaunay Triangulation", in *Proc. of 13th Annual Symposium on Computational Geometry,* pp. 424-426, 1997.

[63] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, "Surface Reconstruction from Unorganized Points", *ACM SIGGRAPH,* pp. 71-78, 1992.

[64] D. D. Morris, and T. Kanade, "Image Consistent Surface Triangulation", *Proceedings of 2000 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 332-338, 2000.

[65] G. Vogiatzis, P. Torr, and R. Cipolla, "Bayesian Stochastic Mesh Optimization for 3D Reconstruction", *Proceedings of 14th British Machine Vision Conference,* Vol.2, pp. 711-718, 2003.

[66] J. H. Park, and H. W. Park, "A Mesh Based Disparity Representation Method for View Interpolation and Stereo Image Compression", *IEEE Transactions on Image Processing,* Vol.15, No.7, pp. 1751-1762, 2006.

[67] D. P. Bertsekas, “Auction Algorithms for Network Flow Problems: A Tutorial Introduction”, Computational Optimization and Applications”, Computational Optimization and Applications, 1, pp.7-66, 1992.

[68] M. H. Hayes, *Statistical Digital Signal Processing,* John Wiley & Sons., New York, NY, USA, 1996.

[69] M. Tekalp, *Digital Video Processing,* Prentice Hall, Upper Saddle River, NJ, USA, 1995.

[70] M. A. Fischler, R. C. Bolles, “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”, *Communications of the ACM*, Vol. 24, pp. 381-395, 1981

[71] D. Nister, “An Efficient Solution to the Five-Point Relative Pose Problem”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 6, pp. 756-770, 2004.

[72] E. Tola, S. Knorr, E. İmre, A. A. Alatan, T. Sikora, “Structure from Motion in Dynamic Scenes”, *Proceedings of 2nd Workshop on Immersive Communications and Broadcast Systams (ICOB 2005)*, Oct. 2005.

[73] T. Thormaehlen and H. Broszio and A. Weissenfeld, "Keyframe Selection for Camera Motion and Structure Estimation from Multiple Views", *Proceedings of 12th European Conference on Computer Vision*, Vol. 1, pp. 523-535, 2004.

[74] M. Pollefeys, “Automatic 3D Modeling with a Hand-Held Camera Images”, *2nd International Symposium on 3D Data Processing, Visualisation and Transmission,* Tutorial Notes, 2004.

[75] P.H.S. Torr, A.W. Fitzgibbon and A. Zisserman, "The Problem of Degeneracy in Structure and Motion Recovery from Uncalibrated Image Sequences", *International Journal of Computer Vision*, Vol. 32 (1), pp. 27-44, 1999.

[76] O. Devillers, S. Meiser, M. Teillaud, "Fully Dynamic Delaunay Triangulation in Logarithmic Expected Time per Operation", *Computational Geometry in Theory and Application,* Vol. 2, No. 2, pp. 55-80, 1992.

[77] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, *2nd Ed.*, Springer Verlag, Germany, 2000.

[78] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, "Optimization by Simulated Annealing", *Science*, Vol. 220, No. 4598, pp. 671-680, 1983.

[79] *Middlebury Stereo Vision Page Data*, http://cat.middlebury.edu/stereo/data.html (last visited on 03.07.2007).

[80] *Image Based Realities- 3D Video Download,* http://research.microsoft.com/vision/InteractiveVisualMediaGroup/3DVideoDownload/ (last visited on 03.07.2007).

[81] C. Çığla, X. Zabulis, A. A. Alatan, "Region-Based Dense Depth Extraction from Multi-View Video", to appear in *Proceedings of 2007 IEEE International Conference on Image Processing (ICIP2007)* 2007.

[82] E. İmre, S. Knorr, B. Özkalaycı, U. Topay, A. A. Alatan, T. Sikora, "Towards 3D Scene Reconstruction from Broadcast Video", *Signal Processing: Image Communication*, Vol. 22, No. 2, pp. 108-126, 2007.

[83] E. İmre, A. A. Alatan, U. Güdükbay, "Rate-Distortion Based Piecewise Planar 3D Scene Geometry Representation", to appear in *Proceedings of 2007 IEEE International Conference on Image Processing (ICIP2007)* 2007.

[84] S. Blackman, R. Popoli, *Design and Analysis of Modern Tracking Systems,* Artech House, UK, 1999.

[85] S. Knorr, E. İmre, T. Sikora, A. A. Alatan, "A Geometric Segmentation Approach for the 3D Reconstruction of Dynamic Scenes in 2D Video Sequences", *Proceedings of 14th European Signal Processing Conference (EUSIPCO 2006)*, 2006.

[86] B. J. Tordoff, D. W. Murray, "Guided-MLESAC: Faster Image Transform Estimation by Using Matching Priors", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 27, No. 10, pp. 1523-1535, 2005.

[87] O. D. Cooper, N. W. Campbell, "Augmentation of Sparsely Populated Point Clouds Using Planar Intersection", *Proceedings of 2004 IASTED Conference on Visualisation, Image and Image Processing*, pp. 359-364, 2004.

[88] A. Bartoli, "Piecewise Planar Segmentation for Automatic Scene Modeling", *Proceedings of 2001 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 283-289, 2001.

[89] *H.264/AVC JM Reference Software,* http://iphome.hhi.de/suehring/tml/download/old_jm/jm95.zip (14.08.2007)

[90] I. E. Richardson, *H.264 and MPEG Video Compression*, John Wiley & Sons, USA, 2003.

[91] G. Taubin, J. Rossignac, "Geometric Compression Through Topological Surgery", *ACM Transactions on Graphics,* Vol. 17, No. 2, pp. 84-115, 1998.

[92] A. A. Alatan, L. Onural, "Estimation of Depth Fields Suitable for Video Compression Based on 3-D Structure and Motion of Objects", *IEEE Transactions on Image Processing,* No. 7, Vol. 6, pp. 904-908, 1998.

[93] Y. Kanazawa, K. Kanatani, "Do We Really Have To Consider Covariance Matrices for Image Feature Points?", *Electronics and Communications in Japan, Part 3*, Vol 86, No.1, pp. 1-10.

[94] *MPEG-4 Description*, http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm, (last visited on 02.09.2007).

[95] *Sebastian Knorr's Homepage- Forschung*, http://www.nue.tu-berlin.de/wer/knorr/sfm.html (last visited on 05.09.2007).

[96] *Marc Pollefeys' Homepage,* http://www.cs.unc.edu/~marc/ (last visited on 05.09.2007).

[97] *MMRG ftp server*, ftp:: \\Mmrgserver\Stereo Data\Yeni Datalar\Stereo\FLOWERPOT (last visited on 05.09.2007).

[98] *Oxford Visual Geometry Group Homepage,* http://www.robots.ox.ac.uk/~vgg/data/data-mview.html (last visited on 05.09.2007).

VITA

PERSONAL INFORMATION

Surname, Name: İmre, Evren

Date and Place of Birth: 23 May 1977, İzmir

E-mail: eimre@metu.edu.tr, e105682@yahoo.com

EDUCATION

- **M. S. (9/2000 – 9/2002)**

  Department of Electrical & Electronics Engineering

  Middle East Technical University (METU), Ankara, Turkey

- **B. Sc. (9/1995 – 6/2000)**

  Department of Electrical & Electronics Engineering

  Middle East Technical University (METU), Ankara, Turkey

- **Minor Programme on Psychology (9/1997 – 6/1999)**

  Department of Psychology

  Middle East Technical University (METU), Ankara, Turkey

WORK EXPERIENCE

- Research and Teaching Assistant (9/2000 – Present)
  - Department of Electrical & Electronics Engineering,

    Middle East Technical University (METU), Ankara, Turkey

- **Research Projects**
  - **Multi-Target Tracking in Video (1/2003-1/2004)**

    Supported by ASELSAN Military Electronics, Ankara, Turkey
  - **Image Feature Based Target Tracking in Video (4/2005-1/2006)**

    Supported by ASELSAN Military Electronics, Ankara, Turkey
  - **3DTV Network of Excellence- Integrated Three Dimensional Television- Capture, Transmission and Display (12/2004-Present)**

    Supported by Europen Comission 6th Framework Inofrmation Society Technologies Programme

PUBLICATIONS

1. **E. İmre**, S. Knorr, B. Özkalaycı, U. Topay, A. A. Alatan, T. Sikora, "Towards 3D Scene Reconstruction from Broadcast Video", *Signal Processing: Image Communication*, Vol. 22, No. 2, pp. 108-126, 2007.
2. **E. İmre**, A. A. Alatan, U. Güdükbay, "Rate-Distortion Based Piecewise Planar 3D Scene Geometry Representation", to be presented at the *2007 IEEE International Conference on Image Processing (ICIP2007),* San Antonio, TX, USA.
3. **E. İmre**, A. A. Alatan, Uğur Güdükbay, "Rate-Distortion Guided Piecewise Planar 3D Scene Representation", presented at the *15th IEEE Conference on Signal Processing and Communication Applications (SIU 2007),* Eskişehir, Turkey, 2007 (in Turkish).
4. **E. İmre**, U. Güdükbay, A. A. Alatan, "Piecewise Planar 3D Reconstruction in Rate-Distortion Sense", presented at the *3D-TV Conference (3DTV-CON 2007*), Kos, Greece, 2007.
5. **E. İmre**, S. Knorr, A. A. Alatan, T.Sikora, "Prioritized Sequential 3D Reconstruction in Video Sequences with Multiple Motions", in *Proceedings of 2006 IEEE International Conference on Image Processing (ICIP2006),* Atlanta, GA, USA, pp. 2969-2972.
6. S. Knorr, **E. İmre**, T. Sikora, A. A. Alatan, "A Geometric Segmentation Approach for the 3D Reconstruction of Dynamic Scenes in 2D Video

Sequences“, presented at the *14th European Signal Processing Conference (EUSIPCO 2006)*, Florence, Italy, 2006.

7. S. Knorr, **E. İmre**, B. Özkalaycı, T. Sikora, A. A. Alatan, “A Modular Scheme for 2D/3D Conversion on TV Broadcast, “ in *Proceedings of 3rd International Symposium on 3D Data Processing*, *Visualization and Transmission (3DPVT 2006)*, Chapel Hill, NC, USA, 2006, pp. 703-710.

8. **E. İmre**, A. A. Alatan, S. Knorr, T. Sikora, “Prioritized 3D Reconstruction in Video Sequences of Dynamic Scenes”, presented at the *14th IEEE Conference on Signal Processing and Communication Applications (SIU 2006)*, Antalya, Turkey 2006 (in Turkish).

9. E. Tola, S. Knorr, **E. İmre**, A. A. Alatan, T. Sikora, “Structure from Motion in Dynamic Scenes”, presented at the *2nd Workshop on Immersive Communications and Broadcast Systems (ICOB 2005)*, Berlin, Germany, 2005.