Graz University of Technology

Institute for Computer Graphics and Vision

Master's Thesis

Video Registration and Depth Propagation for Changing 3D Environments

Stephan Berer

Graz, Austria, February 2012

Thesis supervisor

Univ.-Prof. Dipl.-Ing. Dr.techn. Horst Bischof

Advisors

Dipl.-Ing. Arnold Irschara

Dipl.-Ing. Dr.techn. Stefan Kluckner

Abstract

As a consequence of the enormous progress made in the last few years, modern structure-from-motion (SfM) techniques now allow the reconstruction of large-scale 3D models from hundreds of images in just a few hours. Input images, which are required for the reconstruction process, are easily accessible through Internet photo collection platforms such as Flickr, Google Images and Panoramio. Additionally, recently evolved 3D reconstruction databases, such as Photosynth and Photofly, contain a vast amount of freely available reconstructed 3D scenes. Furthermore, registration and localisation of images from digital cameras are well explored; that being said, using video cameras rather than images provides additional temporal information. The known geometry together with the video information can be exploited to render realistic 3D videos from arbitrary viewpoints. Such 3D videos bring about additional knowledge to the already existent visual information. Occlusion handling, foreground and background extraction and change detection are only a few of the applications that are simplified with this additional depth information.

This master's thesis proposes a novel approach, combining scene models and 2D videos to render such 3D videos. Additionally, a self-calibration of the camera is implicitly incorporated into the registration process; hence, one can easily calibrate any camera by taking a couple of images or a video from an existing 3D model. For this purpose, a standard pinhole camera with a radial distortion model is used. The radial distortion model is estimated with a non-linear optimiser through minimising the re-projection error.

Finally, a comprehensive evaluation of the applied techniques is given on the basis of real self-captured datasets.

Keywords. Structure-from-Motion (SfM), Video Registration, Video Localisation, Photogrammetric Calibration, Direct Linear Transform, 3D Video, Depth Map

Kurzfassung

Aufgrund des enormen Fortschritts von 3D Rekonstruktionstechniken in den letzten Jahren ist es heutzutage möglich ausgedehnte 3D Modelle in wenigen Stunden zu rekonstruieren. Bild basierte 3D Rekonstruktion ist eine bekannte Technik aus der Computer Vision, die aus mehreren überlappenden Bildern ein realistisches 3D Modell berechnet. Die für eine solche Modellierung benötigten Bildermaterialien können sehr einfach aus Internetfotosammlungen wie Flickr, Google Images oder Panoramio geladen werden. Zusätzlich zu den existierenden Fotodatenbanken entstanden kürzlich weitere Plattformen, die solche rekonstruierten 3D Modelle enthalten. Dadurch ist es möglich den 3D Modellierungsschritt zu überspringen und beliebige Modelle direkt aus diesen Datenbanken zu verwenden. Zwei der größten und bekanntesten 3D Rekonstruktionsportale sind Photosynth und Photofly. Ebenso konnte in der Lokalisierung von Kameras anhand von Bildern ein immenser Fortschritt erzielt werden. Durch die Verwendung von Videos, anstelle von einzelnen Bildern, können zusätzlich zeitliche Informationen gewonnen werden. Aus den rekonstruierten 3D Modellen in Kombination mit den Videoinformationen kann ein realistisches 3D Video erstellt werden. Mit der zusätzlich vorhandenen Tiefeninformation in den 3D Videos können Aufgaben wie Vordergrund-, Hintergrund- oder Verdeckungsbehandlung vereinfacht werden.
Diese Masterarbeit beschäftigt sich mit der Erstellung eines solchen realistischen 3D Videos aus einem 2D Video. Der hier vorgeschlagene Ansatz beinhaltet des Weiteren eine Selbstkalibrierung, wodurch jede beliebige Digitalkamera durch die Aufnahme einer geringen Anzahl von Bildern kalibriert werden kann. Um die Genauigkeit der Lokalisierung zu verbessern wurde ein herkömmliches Pinhole-Kamera-Modell mit einem radialen Verzeichnungsmodell erweitert. Die gesuchten Verzeichnungsparameter werden durch Minimierung des Projektionsfehlers unter Zuhilfenahme eines nicht linearen Optimierers geschätzt. Abschließend wird eine umfassende Evaluierung der angewendeten Techniken sowie des vorgeschlagenen Ansatzes anhand von selbst aufgenommenen Datensätzen durchgeführt.

Schlagwörter. Structure-from-Motion (SfM), Video Registrierung, Video Lokalisierung, Photogrammetrische Kalibrierung, Direkte Lineare Transformation, 3D Video, Tiefenkarte

Danksagung

Diese Masterarbeit besiegelt den Abschluss meines Studiums, weshalb ich diese Gelegenheit nützen möchte um allen lieben Personen, die mich während meines gesamten Studiums unterstützt haben von Herzen zu danken. Ihr alle seid dafür verantwortlich, dass ich mein Studium erfolgreich abschließen konnte.

Ich möchte mich an erster Stelle herzlichst beim gesamten Institut für Maschinelles Sehen und Darstellen und bei der Firma Siemens CT bedanken. Allen voran bei Univ.-Prof. Dipl.-Ing. Dr.techn. Horst Bischof für die fachliche Betreuung der Masterarbeit und der Betreuung als Mentor während des Masterstudiums. Des Weiteren bei Dipl.-Ing. Claudia Windisch und ihrem gesamten Team, das diese Arbeit erst ermöglichte. Dipl.-Ing. Arnold Irschara und Dipl.-Ing. Dr.techn. Stefan Kluckner gilt mein Dank für die ausgezeichnete fachliche, kompetente und persönliche Betreuung während des Verfassens dieser Masterarbeit.

Der größte Dank gilt jedoch meiner Familie, welche mich in all meinen Vorhaben und meiner akademischen Laufbahn gefördert und unterstützt hat. Durch euer Verständnis, Motivation und Beistand war es mir möglich, das Studium so zielstrebig und konsequent durchzuziehen - Mama, Papa, Philipp und Jasmin Danke für eure Unterstützung. Nicht vergessen möchte ich meine engsten Verwandten und natürlich meine Großeltern.

Abschließend ein Dankeschön an meine engsten Freunde, die mich auch während des Studiums ständig begleitet haben und für die nötige Abwechslung abseits des Studiums gesorgt haben. Alle hier aufzuzählen würde den Rahmen sprengen, deshalb ein geschlossenes Danke an euch alle.

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am ..

(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

..

date (signature)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Motivation

In the field of computer vision three-dimensional (3D) image-based reconstruction is a method of determining a 3D computer model of arbitrary real-world scenes by taking a bundle of images from various viewpoints [20, 31, 33, 89, 94]. 3D reconstruction is a challenging problem in the computer vision community but a great deal of progress has been made over the last decade due to enormous advancements in 3D scanning technology and the increasing hardware performance of personal computers. Furthermore, the increase in bandwidth of the Internet enables nearly everyone to access and share reconstructed 3D models. As a consequence of these trends new applications, in fields such as e-commerce and virtual museums, have been developed as addressed by Bernardini and Rushmeier [9].

In practice, several technologies, which can be classified into active and passive methods, are available to measure the shape of an object. Active methods interfere with the object itself, while passive methods simply use a camera to measure the light emitted by the object. Examples for active methods range from time-of-flight cameras, triangulation with laser scanners to structured light or modulated light, while image-based rendering is a characteristic passive method. Figure 1.1 shows a 3D model reconstructed with an image-based technique similar to the approach addressed by Snavely et al. [94, 95], which takes as input a number of overlapping images of the same object, captured from various viewpoints and simultaneously computes a 3D model of the underlying scene and additionally estimates the camera pose for each image. This technique is known as structure-from-motion (SfM) since the 3D model is reconstructed from images taken at different viewpoints. Other recently proposed image-based developments are the Photosynth[1] project, mainly driven through Microsoft Research Labs and the Autodesk's Photofly[2] project. Both projects mentioned have a commercial background, therefore, major investments are to be expected.

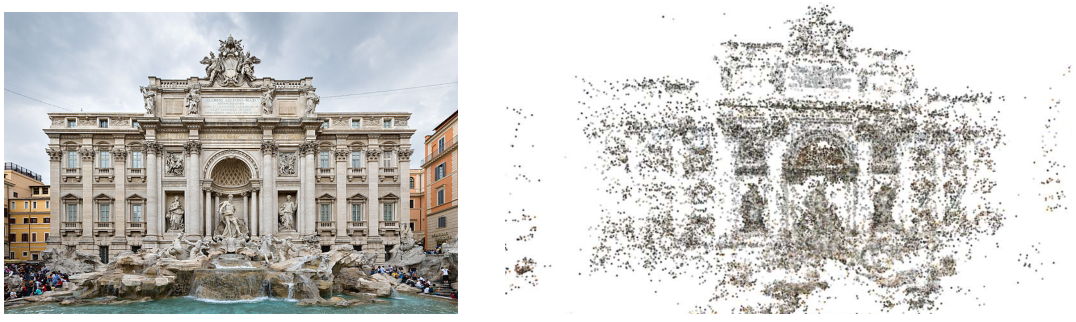

(a) (b)

Figure 1.1: a) Photograph of the Trevi Fountain in Rome. b) 3D model of the same object, reconstructed by applying an image-based approach as described in [94, 95]. These images are taken from Wikipedia[3] and [94], respectively.

In contrast to the passive methods, Figure 1.2 shows a reconstructed model using the triangulation principle and a standard laser scanner, which is an active method. Levoy et al. [52] developed equipment for such a 3D laser scanner and used it to scan and reconstruct Michelangelo David's statue. Another up-to-date passive method is Microsoft's Kinect camera, usually constructed as a novel interface for video gaming; though, it can also be used for depth measurements and 3D reconstruction. This consumer depth camera has caused a revolution in the computer vision community since it is the first affordable depth camera. Another circumstance, which shows the immersive interest in this new camera technique, is the fact that Microsoft sold approximately eight million Kinects in just the first two months, as Microsoft announced at 2011 International CES. The KinectFusion [47, 48, 72], as Microsoft's 3D reconstruction project is called, can reconstruct entire scenes in real-time using a novel GPU-based approach. An entire room, as shown in Figure 1.3, can be reconstructed in only a few seconds by moving the Kinect camera around in the room.

[1]http://photosynth.net
[2]http://labs.autodesk.com/technologies/photofly/
[3]http://en.wikipedia.org/wiki/Trevi_Fountain

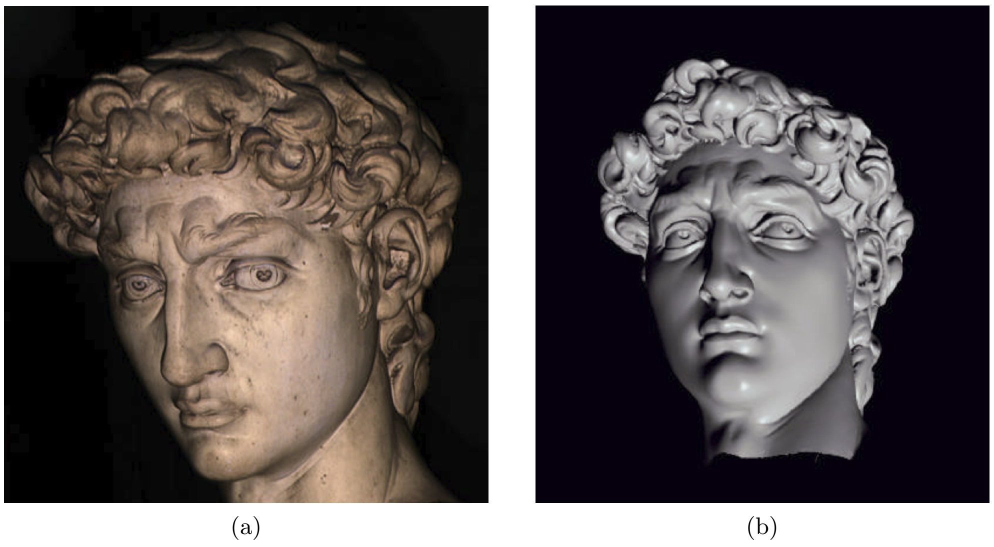


Figure 1.2: a) Photograph of the head of Michelangelo's David. b) 3D model of the same object, reconstructed by using a laser scanner and triangulation principle. These images are taken from [52].

Moreover, finding the position of a mobile device can be interesting for many applications, such as tourist guiding or augmented reality. Nowadays, most mobile devices have a Global Positioning System (GPS) module included; however, the receivers do not always work inside buildings or in environments where the sky is occluded. One simple solution to overcome this problem is image-based approaches, which are, on the one hand, more accurate than the position determined by the GPS and, on the other hand, are applicable in both indoor and outdoor environments. The ability to determine the pose of an image, with respect to a 3D model, is a highly relevant research topic [46] which has made a great deal of progress in the last decade. There is an enormous amount of literature available about both image registration and localisation [92]. Gordon and Lowe [34] proposed an approach similar to ours by matching feature descriptors to determine the camera pose. Robertson and Cipolla [82] match images, which are captured with a mobile device, against a database of views to estimate the pose of the camera. Another similar project, which utilises databases as well but focuses on urban environments and the estimation of the GPS position, is addressed by Zhang et al. [121]. Schindler et al. [87] proposed a novel location recognition method introducing a vocabulary tree. The work by Oskiper et al. [75] uses a similar approach, focusing on real-time registration. Simultaneous object localisation and camera estimation on a pre-computed 3D model is explained in [117]. Sattler et al. [50] proposed a novel direct 2D-to-3D matching approach that speeds up the localisation time enormously. Since standard cameras have some sort of distortion included, Josephson and Byrod [50] incorporated a radial distortion model directly into the random sample consensus

(RANSAC) step bringing about more accurate outcomes. Recently, Arth et al. [4] proposed a novel real-time self-localisation approach from mobile panoramic images by transferring a GPS prior to a remote server that contains an offline reconstructed urban model.

Furthermore, the rising Internet capacity has led to extremely large Internet photo collections. Additionally, this trend together with the increasing sever storage capacity gave rise to Internet video collection platforms, such as youtube[1] and myvideo[2]. These platforms allow users to easily share and access videos, such as those of myriad landmarks from around the world.

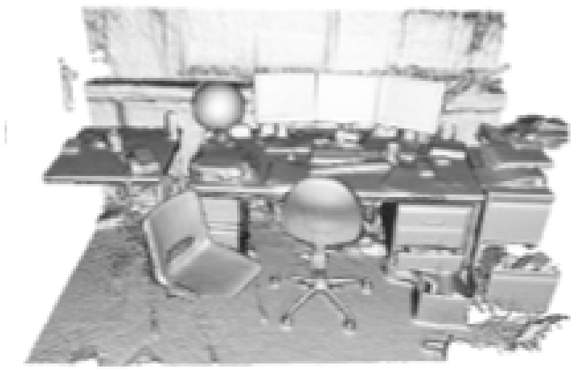


Figure 1.3: 3D reconstruction using Microsoft's Kinect camera, which can reconstruct whole scenes in real-time with just one single hand-held camera. This image is taken from [48].

To summarise, 3D models are available everywhere and can easily be computed with several techniques such as SfM. Furthermore, they can also be captured directly with consumer depth cameras. Additionally, two-dimensional (2D) videos can be found anywhere on the World Wide Web. However, real 3D videos are not yet widely available, but are of commercial and academic interest. Through this additional depth information the difficulty about occlusion is simplified, since the depth defines which objects are in the foreground and which ones are in the background. This is especially interesting in the field of augmented reality. Furthermore, using videos instead of images leads to additional temporal information. Background and foreground modelling is simplified as well, as the reconstruction process only reconstructs rigid objects. The additional depth information allows the detection of foreground objects, such as persons or vehicles, in a simpler way. The same principle is applicable for detecting changes in the videos, too. In addition, regions without distinctive characteristics (i.e. sky) can be detected and later extracted from the videos.

[1]http://www.youtube.com
[2]http://www.myvideo.de or www.myvideo.at

Since 3D television sets have become affordable, several semi-automatic and fully automatic approaches to converting existing 2D video content into 3D videos have been proposed in recent years. Fully automatic methods require no user interaction, while in contrast semi-automatic approaches need some kind of user input to assign depth values to the video frames. As it is not possible to recover a scene geometry from one position in the image plane, as explained by Brosch et al. [11], fully automatic approaches must include additional information. The work addressed by Moustakeas et al. [70] utilises motion to solve this problem. The approach proposed by Zhang et al. [120] synthesises a 3D video directly from the motion of a 2D video by selecting various frames from the video sequence instead of pre-computing a depth map.
Contrarily, semi-automatic methods require user interaction. Generally, the user assigns some depth values to pre-segmented regions or even to each pixel in one or more video frames. This depth information is further propagated through the video frames. The work proposed by Wu et al. [116] assigns depth values to pre-segmented regions, while Li et al. [57] and Varekamp and Barenbrug [109] define depth values for each pixel in several keyframes. Brosch et al. [11] and Guttmann et al. [35] propagate sparse user-provided depth information, termed scribbles, over the entire monocular video.

In conclusion, the 3D reconstruction process and the modern localisation approaches enable, along with the Internet image and video platforms, the registration of videos onto 3D models. Hence, the main focus of this master's thesis is to extend the base approaches of registering images to a more sophisticated approach of video registration. The aim is to take a bundle of images, reconstruct a 3D model and subsequently register a video onto this model. After finding homogenous regions in the video the depth for each region can be estimated, which makes it possible to render realistic 3D videos and accurate depth-map videos from arbitrary viewpoints. Moreover, the registration process estimates the intrinsic camera parameter for each video frame separately; therefore calibration of arbitrary cameras is implicitly embedded. As a consequence, camera parameters of arbitrary images and videos from the web can be estimated without further knowledge about the camera type or focal length.

1.2 Introducing the Overall Workflow

This section outlines the aim of this master's thesis and introduces the main workflow. Figure 1.4 gives a brief overview of the major parts.
First of all, our approach requires a 3D model, which is in our case computed with a state-of-the-art SfM workflow [45, 81, 94]. Each 3D point holds information about the position, scale and colour as well as a feature descriptor, which is required during the localisation process. The next part deals with the image and video view registration and particularly with camera pose estimation [30, 46, 56, 84, 123]. A 6-point direct linear transform (DLT) algorithm [39, 118] included into a RANSAC framework [25] calibrates the camera and

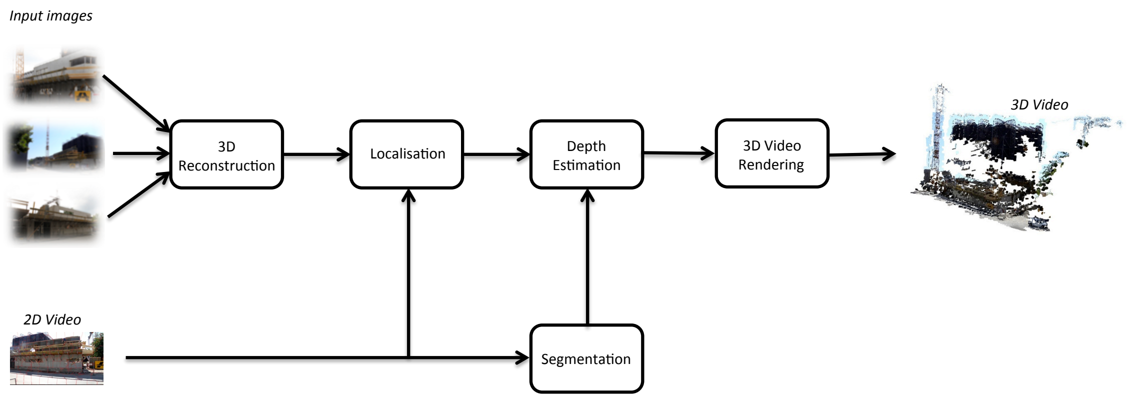


Figure 1.4: Brief overview of the 3D video generation workflow.

additionally estimates the camera pose. Further, the 6-point DLT is substituted by a pre-calibration step and a 3-point algorithm resulting in more efficient computation [36]. Moreover, to get consistent regions the video frames are segmented with a state-of-the-art segmentation algorithm [24, 55, 62, 69, 111]. Finally, the depth information from the 3D model is projected back into the registered video and, as a consequence of this projection and a robust estimation, each segmented region is assigned a depth value. This additional depth information leads to a realistic 3D video.
A comprehensive explanation describing the methods used in more detail is given in Chapter 4, which also includes some sample applications and a flow chart to improve the understanding of the overall workflow. The workflow can be summarised as follows:

1. Capturing or downloading images and videos
2. Computing a 3D model from the images or downloading a pre-computed model
3. Video localisation using 6-point DLT included into a RANSAC framework
4. Non-linear optimisation of the radial distortion parameter
5. Segmenting the video frames into consistent regions
6. Projection of the 3D points onto the segmented video frames to get depth information
7. Rendering 3D video from the pre-assigned depth values

1.3 Structure of the Master Thesis

This chapter has already extensively introduced the motivation and the aims of this master's thesis.
Chapter 2 gives an overview of the basic methods regarding 3D modelling. First, some

general information about the 3D reconstruction process is given. Second, some basic background information, which is necessary to understand the 3D reconstruction process, is addressed. Finally, this section explains the main image-based techniques used to reconstruct sparse and dense 3D models, respectively.
Chapter 3 introduces the methods and techniques required to register a single video frame onto a pre-computed 3D model. Furthermore, this chapter gives an introduction to camera calibration including image distortion models. Additionally, the estimation of the radial distortion parameters through a non-linear optimisation is addressed in this chapter.
In Chapter 4 the entire workflow is presented; explaining the interaction of the various parts of the 3D video registration pipeline. Furthermore, the 3D rendering process is described and some sample applications are illustrated.
Chapter 5 evaluates our approach in terms of accuracy and robustness. Several results on images and video sequences are presented is this chapter. Finally, a summary of the obtained results is given.
Finally, Chapter 6 includes final conclusions, addresses some open issues and gives an outlook on future work.

Chapter 2

3D Modeling

Contents

This chapter gives an overview of the basic methods regarding 3D modelling. First, some general information about the 3D reconstruction process is given, the image-based approach, in particular, is explained in detail. Second, some basic background information, which is necessary to understand the 3D reconstruction process, is addressed. This includes the topics of pinhole cameras, epipolar geometry and image-based 3D reconstruction utilising the triangulation principle. Next, this section explains the main SfM techniques used to reconstruct sparse and dense 3D models, respectively. Additionally, a sparse and a dense 3D reconstruction model computed with our workflow are presented.

2.1 General 3D Reconstruction Methods

As addressed in the introduction, several 3D reconstruction techniques including time-of-flight cameras, structured light, 3D laser scanners, image-based methods, etc. have been developed in the last few decades. Figure 2.1 classifies the various approaches into contact and non-contact methods and, additionally, gives an overview of the best known techniques. Non-contact methods can be further divided into active and passive methods. Grey shaded boxes represent some of the most well-known 3D reconstruction methods, while the red box illustrates an image-based modelling approach. SfM is one of the most popular image-based modelling approaches and is used as the reconstruction method throughout this master's thesis.

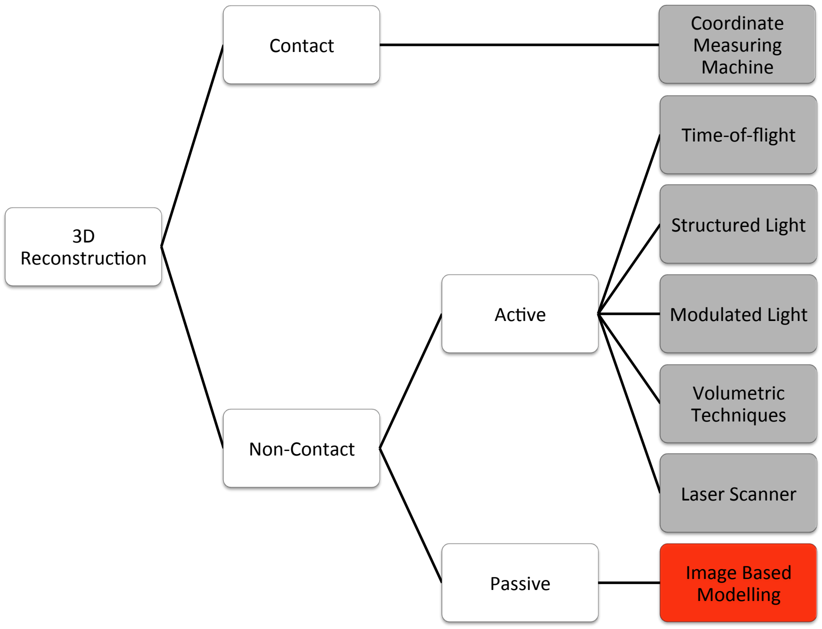


Figure 2.1: Classification of 3D reconstruction techniques into contact and non-contact methods. The red box illustrates an image-based modelling approach, which is used as the 3D reconstruction method throughout this master's thesis.

SfM is an approach that takes a number of overlapping images of the same object, captured from various viewpoints and automatically computes a 3D model of the underlying scene and additionally estimates the camera pose for each image. SfM has been a challenging problem in the computer vision community for the last 25 years, but a great deal of progress has been made over the last decade due to advances in digital imaging technology. With the premises mentioned in the introduction it is now regarded as unsophisticated to create such 3D models as state-of-the-art SfM approaches can reconstruct such large-scale 3D models in only a few hours [2, 44, 45, 78, 81].

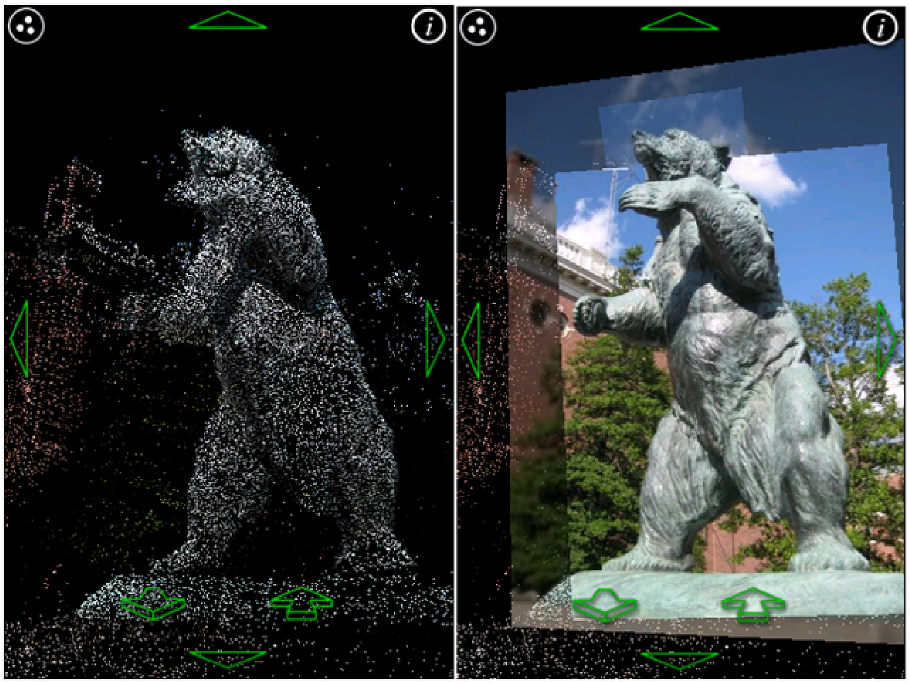


Figure 2.2: Photorealistic 3D reconstruction using Microsoft's Photosynth[1] approach.

As addressed in the introduction, one of the first and most well-known SfM approaches is the Photo Tourism project implemented by Snavely et al. [94]. This Photo Tourism project resulted in the *Photosynth* development, predominately supported by Microsoft, which takes as input a number of images, as the Photo Tourism project does, and stitches these images together to render a photorealistic 3D model. The Photosynth project allows users to upload their own panoramas as well as create their own 3D visualisations through a desktop application. Nowadays, mobile applications, for various mobile platforms, are available that allow users to stitch images together and create their own visualisations directly on their mobile devices. A visualisation utilising this approach is shown in Figure 2.2.
Another similar project is the *Photofly*[3] project, developed by Autodesk Labs, which, in contrast to the Photosynth project, uses cloud computing to render photorealistic 3D models. Additionally, Photofly enables measurements inside the 3D model and the adding of arbitrary geometry to the model. A person reconstructed with the Photofly toolbox is shown in Figure 2.3. On top of this project Labatut et al. [16] proposed a novel method which can reconstruct 3D models acquired through passive stereo techniques.

[1]Image is taken from http://blogs.msdn.com
[2]Image is taken from http://www.flickr.com/photos/btl/5762942031/
[3]http://labs.autodesk.com/technologies/photofly/

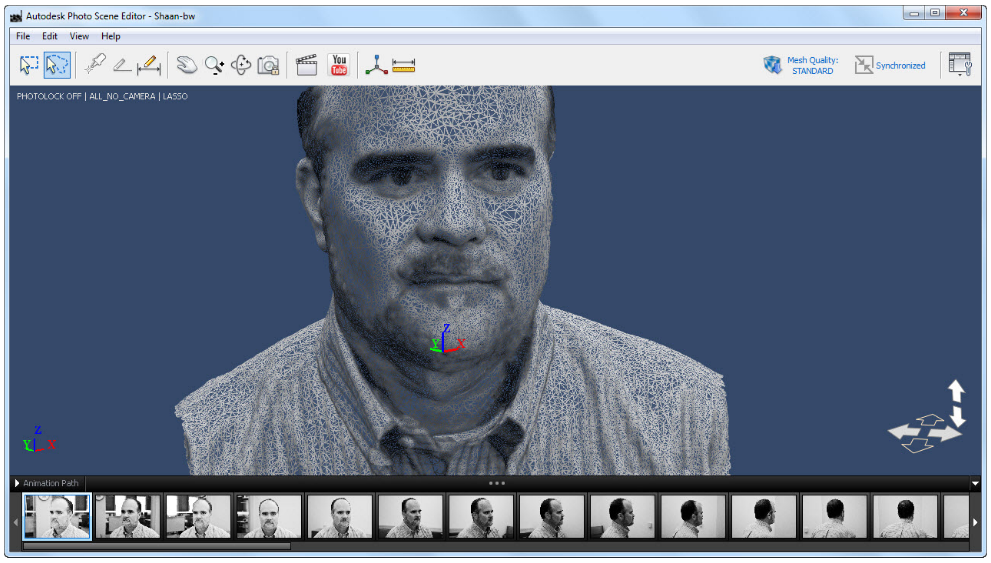

Autodesk Photo Scene Editor - Shaan-bw
File Edit View Help
Mesh Quality: STANDARD
Synchronized
PHOTOLOCK OFF | ALL_NO_CAMERA | LASSO
Animation Path

Figure 2.3: 3D reconstruction using Autodesk's Photofly approach. Several images from various viewpoints (see bottom area of this image) from the same person are used to reconstruct an accurate 3D model[2].

2.2 Theoretical Background

In this section some mathematical background information regarding Pinhole cameras, epipolar geometry and image-based 3D reconstruction are addressed, since these techniques are fundamental for the generation of 3D representations.

2.2.1 Pinhole Camera Model

In general, a camera defines a mapping from 3D world coordinates to corresponding 2D image coordinates. Figure 2.4 shows the basic geometry of an ideal pinhole camera, which projects a 3D point $\mathbf{X}$ onto a 2D point $\mathbf{x}$ on the *image plane*. The origin of the camera coordinate system is equal to the centre of the camera (C), which is also known as the *projection centre* or *optical centre*, respectively. The X and Y axes of the camera coordinate system span a plane which is parallel to the image plane and perpendicular to the Z axis of the camera coordinate system. As per the definition, the Z axis of the camera coordinate system points in the view direction of the camera and is termed *principal axis* or *optical axis*. Considering Figure 2.5, one can observe that the mapping from a scene point $\mathbf{X} = (x, y, z)^T$ to an image point $\mathbf{x}$ depends only on the focal length f of the camera, hence we can define the *perspective projection* as

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \rightarrow \begin{pmatrix} \frac{fX}{Z} \\ \frac{fY}{Z} \end{pmatrix}. \tag{2.1}$$

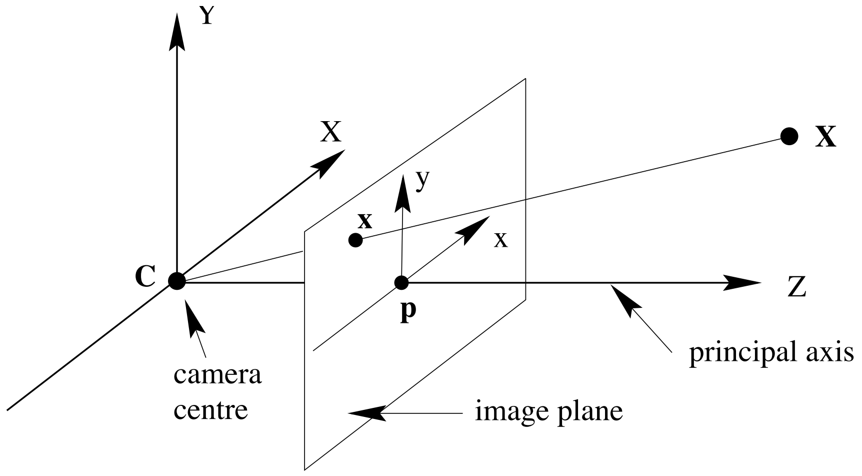


Figure 2.4: Three dimensional illustration of the ideal pinhole camera geometry taken from [39]. A point **X** is projected to a point **x** on the image plane. The image plane is, in this case, located in front of the camera centre. The centre of the camera is equivalent to the origin of the camera coordinate system.

Another important term is known as the *principal point* (**p** in Figure 2.4), which is the intersection point between the principal axis and the image plane, and defines the origin of the image coordinate system. To express the mapping from **X** to **x** in Equation (2.1)

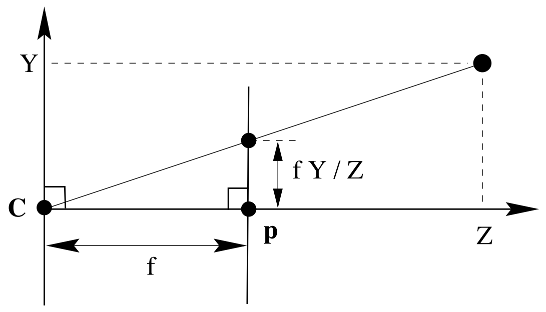


Figure 2.5: Illustration of the pinhole camera geometry in two dimensions. The distance between the camera centre and the image plane (focal plane) is known as the focal length f. The figure is taken from [39].

in a simpler way we introduce *homogenous coordinates* and as a consequence we can write the same equation in the form of a matrix multiplication

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{pmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.2}$$

where f is the focal length of the camera. To simplify further notations the matrix in the previous equation can be written as diag(f,f,1)[I|**0**]. Equation (2.2) can be compactly expressed by writing

$$\mathbf{x} = P\mathbf{X}, \tag{2.3}$$

where $\mathbf{X} = (X, Y, Z, 1)^T$, $\mathbf{x} = (x, y, 1)$ and P=diag(f,f,1)[I|**0**].

So far, we have assumed that the origin of the image coordinate system is at the bottom left corner of the image, but in general this is not the case; therefore, in addition to the previous definition we are introducing a more generic camera model by adding a *pixel skew* s and a *principle point offset* (p_x, p_y), which defines the translation between the camera coordinate system and the image coordinate system as illustrated in Figure 2.6. Now we can define a new matrix

$$K = \begin{pmatrix} f & s & p_x \\ & f & p_y \\ & & 1 \end{pmatrix}, \tag{2.4}$$

which contains the focal length f, the principal point offset (p_x, p_y) and the pixel skew s. This matrix is known as the *camera calibration matrix* and contains the *intrinsic* camera parameter. Finally, a more generic model will be introduced, which allows camera *rotation*

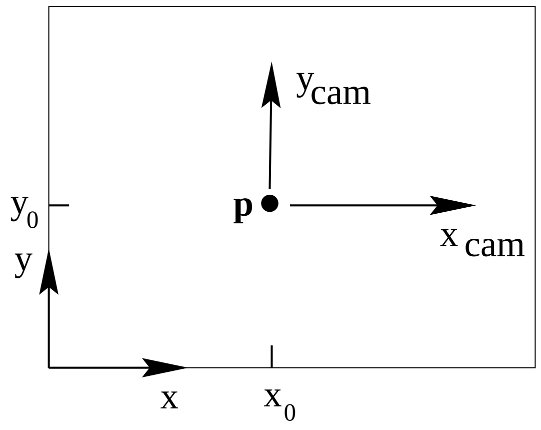


Figure 2.6: The translation between the camera (x_{cam}, y_{cam}) coordinate system and the image (x, y) coordinate systems is defined by the principal point offset $(p_x, p_y) = (x_0, y_0)$. This image is taken from [39].

and *translation*. This model, which is used throughout this thesis, can be expressed by

$$P = K[R|\mathbf{t}], \tag{2.5}$$

where $\mathbf{t} = -RC$. R represents a 3×3 rotation matrix and C is equal to the position of the camera centre referring to the world coordinate system [39].

Moreover, we should mention that the projection matrix P has eleven degree of freedom (DOF) hence six point correspondences are necessary to estimate the projection matrix explicitly, since one correspondence solves two equations. If the intrinsic camera parameters are well known and only the extrinsic camera parameters (rotation and translation) are unknown the matrix only has six DOF, therefore three point correspondences are enough for estimation of the camera pose [39].
More information about camera models and detailed explanations about central projection can be found in [39] and in [99], which uses the same camera model. For purposes of accuracy a radial distortion model is introduced which is explained in more detail in Section 3.3.2.

2.2.2 Epipolar Geometry

Epipolar geometry is fundamental to multiple view geometry, since it defines the relation between two image planes. It is mainly driven through the *epipolar constraint*, as it reduces the correspondence problem from a 2D search to a 1D search on the epipolar line.

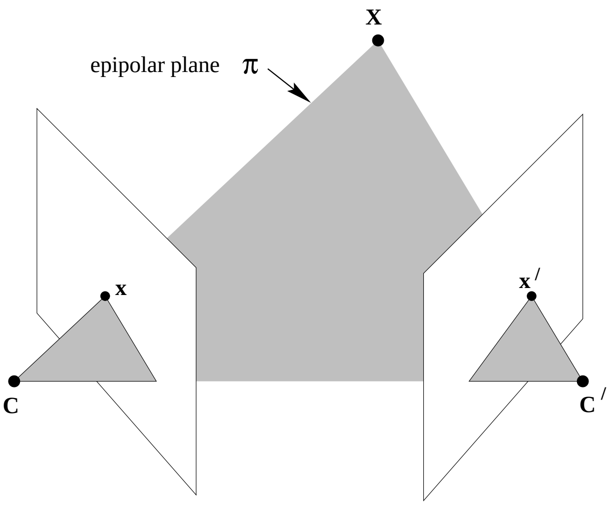


Figure 2.7: Epipolar geometry. The two camera centres, represented by **C** and **C**', span together with the point **X** the epipolar plane π. The image is taken from [39].

As shown in Figure 2.7 the camera centres (**C** and **C**'), the image points **x** and **x'** and the 3D point **X** are coplanar. This plane, denoted as π, is called the *epipolar plane*. Another important term is the *baseline* joining the two projection centres. It is noted that every point lying on the ray from the projection centre **C** to the space point **X** is projected to the same point **x** in the image plane. From the coplanarity we can derive the definition of the *epipolar line*, which is the intersection between the image plane and the epipolar plane, as illustrated in Figure 2.8. The epipolar line is defined as **l'** and joins the *epipole* **e'**, which is the intersection between the baseline and the image plane. In other words, the epipole is the projection from the first camera centre into the second image plane and vice versa [39].

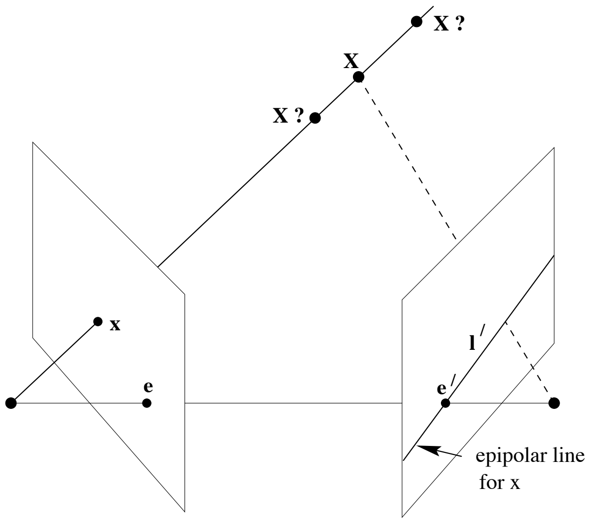


Figure 2.8: The epipolar constraint defines that point **x** in the first images lies on a line **l'** in the second image. This image is taken from [39] (modified).

From the previous definitions we can derive the epipolar constraint, which states that a point in the first image is projected into a line, known as an epipolar line, in the second image and vice versa. By applying the epipolar constraint, the correspondence problem is reduced from a 2D to a 1D searching problem, resulting in a speed up in the matching process. The epipolar constraint is visualised in Figure 2.9. A point (white dot) in the first image (Figure 2.9a) corresponds to a white line (epipolar line) in the second image (Figure 2.9b) and vice versa [39].

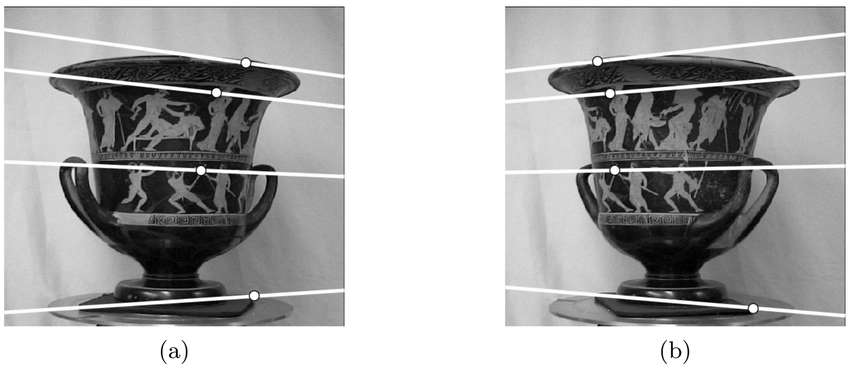


Figure 2.9: An epipole (white dot) in image a) correspond to an epipolar line (white line) in image b) and vice versa. These images are taken from [39].

Loop and Zhang [59] proposed a new technique, called rectification, to additionally speed up the searching process by transforming the image in such a way that the epipolar lines become horizontal. After applying this approach the correspondence problem is solved by finding a match along a horizontal line, which greatly simplifies the implementation and speeds up the matching process enormously.
Mathematically the previous geometric explanations can be described in what is known as the fundamental matrix F, which denotes

$$\mathbf{x} \mapsto \mathbf{l}'. \tag{2.6}$$

For corresponding points **x** and **x'** the fundamental matrix F is defined as

$$\mathbf{x}'^T F \mathbf{x} = 0. \tag{2.7}$$

From Equation (2.7) we can derive the epipolar lines, where

$$\mathbf{l}' = F\mathbf{x} \tag{2.8}$$

and

$$\mathbf{l} = F^T \mathbf{x}', \tag{2.9}$$

additionally the epipoles

$$F\mathbf{e} = \mathbf{0} \tag{2.10}$$

$$F^T \mathbf{e}' = \mathbf{0}. \tag{2.11}$$

More information about multiple view geometry and especially epipolar geometry can be found in [39] and in [99].

2.3 Image-based 3D Reconstruction

First of all, the reconstruction process can be classified into *metric, Euclidian and projective reconstruction* depending on the known camera parameter. At this stage we assume the correspondence problem is solved and a number of correspondences are given. The user friendliest way to reconstruct a 3D model is the projective reconstruction method, which makes no assumption of the intrinsic and extrinsic camera parameters. If the intrinsic camera parameters are known, a Euclidian reconstruction can be applied, which reconstructs the 3D model up to an undefined scale parameter. Through self-calibration and localisation the camera matrixes P and P', which include the intrinsic and extrinsic camera parameters, can be estimated [39].
If we assume the correspondence problem is solved and two corresponding points **x** and **x**', which satisfy Equation(2.7), as given then a simple triangulation can be applied to re-

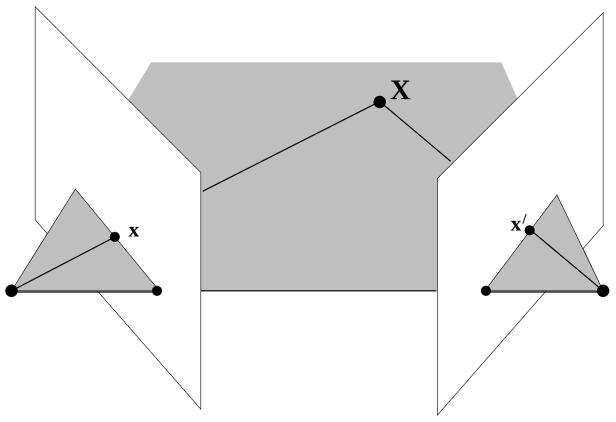


Figure 2.10: Illustrates the triangulation principle. If Equation (2.7) is satisfied, the rays from the camera centres to **x** and **x**', respectively, span one plane, hence the rays must intersect in one 3D point **X**. The image is taken from [39].

construct the 3D point **X**, as shown in Figure 2.10. Through the epipolar constraint, both image points and both camera centres lie in the same plane, hence both rays must intersect at one point **X**. This triangulation principle only holds when the epiploar constraint (see Equation (2.7)) is satisfied, which is generally not the case, due to inaccuracies in the measurements. To solve this problem simple linear triangulation methods can be used, but if accuracy is important more sophisticated approaches, such as maximum likelihood estimator (MLE) should be applied [39].
More details about this 3D reconstruction process, and in particular the triangulation process, are given in [80] and [39].

2.3.1 Sparse Geometry

As a consequence of the progress in the SfM techniques there are now several image-based reconstruction approaches. One of the first SfM methods was invented by Longuet-Higgins [58] in 1981. Later on, techniques such as factorization [101], bundle adjustment [105] and self-calibration [76, 77] had an enormous impact on the progress of SfM techniques.
A more recent reconstruction approach focuses on efficiently matching feature descriptors along multiple images, as addressed by Schaffalitzky and Zisserman [85]. Vergauwen and Van Gool [112] released a web-based application to reconstruct a model of cultural heritages. Through a web interface users can upload a set of images from a scene to a public server. The fully automatic reconstruction process runs on several computers, which are connected to a server. After processing, the reconstructed model can be downloaded through an FTP sever and visualised with a standard viewer. In contrast, Fitzgibbon et al. [26] and Nister [73] started with a bottom-up approach by first matching small subsets and then merging these subsets together into a complete model. Another popular approach is the Bundler implementation developed by Snavely et al. [94, 95], which is especially

useful for unordered image collections. This implementation, which is written in C/C++, was the first one to perform well on a large set of Internet image collections. Recently, Irschara et al. [43] proposed an approach utilising orientation and position priors.

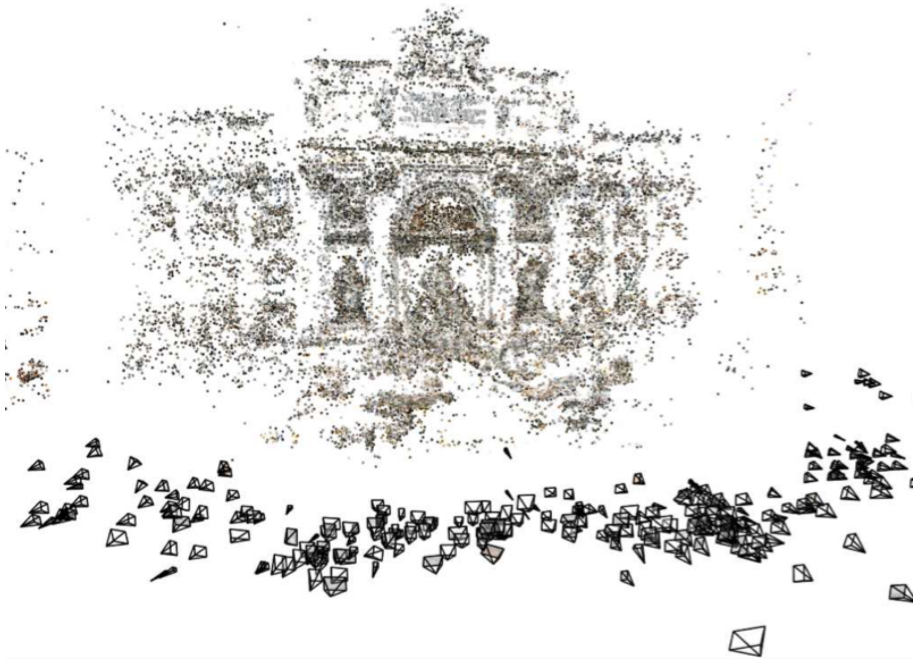


Figure 2.11: Reconstruction of the famous Trevi Fountain in Rome, Italy containing a 3D point cloud and additional estimated camera positions and directions are visualised. This image is taken from [95].

Due to the modularity of our approach which allows us to exchange different parts of the workflow without any influence on the final outcome any of the previously described approaches could be used to reconstruct a 3D model; however we decided to use Bundler, since the source code is publicly available and it performs well on big datasets. Bundler is especially useful for the creation of large-scale architectural models, as addressed in the article. Another benefit is its performance, since it can reconstruct a dataset consisting of approximately 100 images (the number our datasets consist of) in a few hours.
In general SfM approaches take any uncalibrated images as input and can then compute a sparse 3D model incrementally as illustrated in Figure 2.11. Additionally, the relative camera pose for each registrable image is estimated and is also visualised in this figure. The project "Building Rome in a Day" depends on the fundamental implementations of the Bundler software and aims to reconstruct famous landmarks of cities in less than one day through clustering and parallel computing [1, 2]. It mainly bases itself on the implementation of Brown and Lowe [13], however some modifications are included to increase the robustness of the entire system. For instance, the EXIF information of the focal length f, if provided by the image, is used for initialisation. These EXIF values are not precise,

but provide a rough estimate for the purpose of initialisation.
The workflow of a generic SfM pipeline, can be roughly described as follows

1. Find feature points in each image
2. Extract feature descriptor for each feature point
3. Match corresponding feature descriptors
4. Recover 3D model and estimate camera pose through minimising the re-projection error
5. Repeat the previous procedure by adding one more image in each iteration

Furthermore, small baseline stereo and wide baseline stereo methods are distinguished. Since in small baseline approaches the difference between two images is marginal the correspondence problem is simplified. In contrast, in the case of wide baseline techniques it is harder to find correspondences between the images. On the other hand a benefit of wide baseline methods is the large triangulation angle, therefore the depth estimation is more accurate compared to small baseline approaches, which result in a small triangulation angle.

2.3.2 Densification

Since the approach addressed in Section 2.3.1 leads to sparse models this section aims to discuss approaches to reconstruct denser models, as these models are more appropriate for purposes of visualisation. Instead of estimating a depth value for the extracted feature points only, as is done in sparse 3D modelling, a densification approach determines a depth value for each pixel visible in the images. For this purpose the entire geometry must be known, as it greatly simplifies the problem of finding correspondences; therefore, a sparse model is computed first, as addressed in Section 2.3.1.
Applications utilising this technique range from metric reconstruction for engineering and scientific purposes to realistic scene reconstruction for the movie and video gaming industry. According to Furukawa and Ponce [31] densification approaches can be coarsely classified into

- Voxel-based
- Deformable polygonal meshes
- Multiple depth maps
- Patch-based

approaches.
Jiang et al. [49] recently proposed a novel two-step method for 3D dense reconstruction,

which exploit seed points. The work by Strecha et al. [96] focuses on depth estimation from multiple wide-range images, while Van Meerbergen et al. [108] require a small baseline between the images. Furukawa and Ponce [30] simultaneously calibrate the camera and reconstruct a dense model for multiple images by applying an additional bundle adjustment step. Other approaches utilise voxel carving [90], photo hulls [93] and level sets [22]. Recently, Goesele et al. [33] proposed a reconstruction approach which takes as input a set of images from Internet photo collections and directly computes a dense model. The work proposed by Irschara et al. [44] utilises a vocabulary tree to speed up the reconstruction process. To summarise, Scharstein and Szeliski [86] published a taxonomy and comprehensive evaluation of state-of-the-art dense stereo algorithms.

Due to the progress of these image-based modelling approaches several applications concentrating on reconstruction of large-scale architectural models have arisen. For example, the MIT City Scanning project [100] reconstructed a rigid model of the MIT campus by taking a dataset of several calibrated images, acquired in an outdoor region spanning several hundred meters. Another dense reconstruction project driven by the Stanford University [83] used an interactive system of multi-perspective images of a nearly planar scene for visualising urban landscapes and city blocks. This approach takes as input a sideways-looking video, which could be captured for instance from a moving vehicle, and creates multi-perspective strip images, which are later used for reconstruction purposes. Akbarzadeh et al. [3] reconstruct urban 3D models from videos and additionally geo-reference these models. The 4D cities project [88] aims to reconstruct Atlanta from historical images. The approaches addressed here are only a small selection from this wide-ranging research field, but some of the most successful ones. A comparison of recently developed dense reconstruction algorithms and an evaluation of these algorithms was published by Seitz et al. [89].

Due to the modularity of our approach any of these 3D dense reconstruction methods can be applied. As addressed in Section 2.3.1, to reconstruct a sparse model the Bundler method is used; therefore it makes sense to utilise methods which build upon these previous sparse techniques.

For densification of sparse models Bundler proposes two approaches. The first is the patch-based multi-view stereo (PMVS) [31] method and the second is the clustering views for multi-view stereo (CMVS) [29, 31] approach both of which are described in the following section.

2.3.2.1 Patch-based Multi-view Stereo

Although several researchers have published various densification methods over the past few years, we decided to use the PMVS /CMVS approach developed by Furukawa et al. [29, 31], since it cooperates perfectly with the Bundler approach, though all other approaches to compute dense models would work just as well. This approach takes as input a sparse set of matched feature points and outputs a dense model consisting of minor patches. Through simple global visibility tests false matches are filtered leading to a set of visible patches. A benefit of this algorithm is that it detects and discards outliers automatically and no initialisation is required. Figure 2.12 shows the overall process of

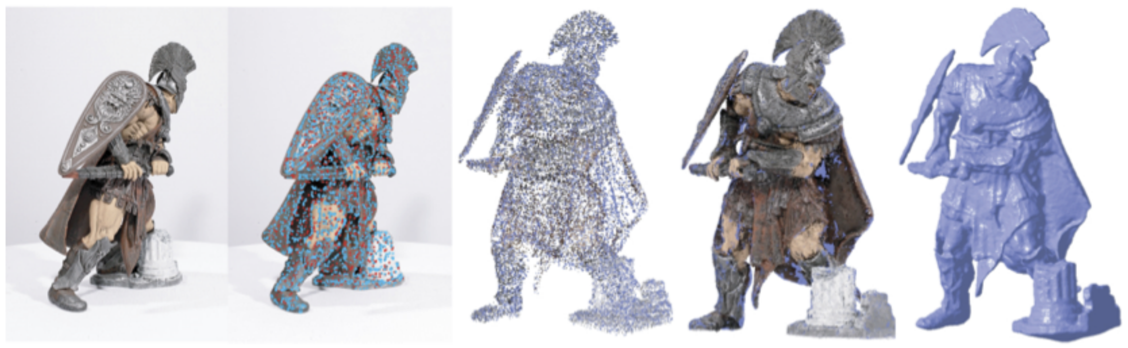


Figure 2.12: Illustration of the entire patch-based multi-view stereo reconstruction process. From left to right: Input image, Harris and difference- of-Gaussians features, reconstructed patches, filtered patches, final mesh model. This set of images are taken from [31].

the PMVS reconstruction process, which is ordered as a simple match, expand and filter procedure [31].
First, Harris and difference-of-Gaussian features are extracted and matched along multiple images, which results in a sparse 3D model. Then neighbouring pixel of the initial matches are used to get a more dense model and finally, through a filtering process, the visibility of each patch is checked to remove inconsistent matches. The last two steps, expansion and filtering, are repeated n times to increase the number of dense patches and to remove false matches.
Additionally, this procedure leads to orientated patches instead of single colour points, which can be useful for several other applications and enhance the visual impression. As addressed by Furukawa, through a hybrid approach the PMVS method is applicable for various input datasets, such as objects and crowded scenes. Another benefit of this approach is its performance. As further addressed by Furukawa according to the Middlebury benchmark test this method outperforms, in four out of six cases, other state-of-the-art approaches in terms of accuracy and completeness. These advantages, together with its availability as freeware, had a major impact on our decision for PMVS. In addition, the CMVS approach [29] builds upon SfM and PMVS respectively. We decided to use the CMVS, because of the previously addressed reasons, though other densification approaches could be used as well. The main goal of this CMVS technique is to decompose similar input images into clusters with a slight overlap and subsequently apply a standard SfM algorithm to each cluster. This principle is shown in Figure 2.13. The advantage of this method is that all clusters are independent; hence the reconstruction process can be parallelised, which results in an enormous speed up in the reconstruction process. The

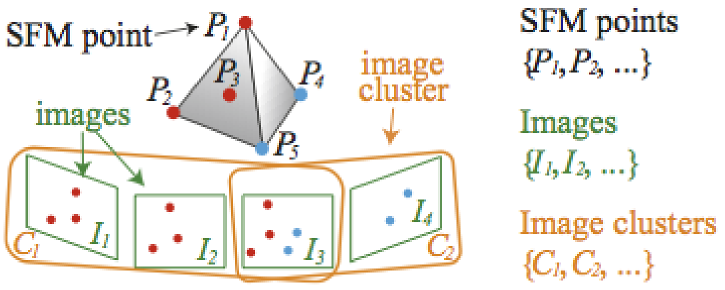

SFM point
P_1
P_2
P_3
P_4
P_5
images
image cluster
I_1
I_2
I_3
I_4
C_1
C_2
SFM points $\{P_1, P_2, \ldots\}$
Images $\{I_1, I_2, \ldots\}$
Image clusters $\{C_1, C_2, \ldots\}$

Figure 2.13: Illustration of the clustering principle. A number of similar images are grouped into one cluster and each cluster is, subsequently, individually reconstructed, instead of reconstructing the entire model from all available images. Afterwards, the reconstructed subsets are combined into one final 3D model. This image is taken from [29].

resulting reconstructed subsets are subsequently merged into one model. Applying this method a 3D model of an enormously large number of image sets can be reconstructed in a reasonable time in comparison to standard SfM techniques, which use as input the entire image set.

2.3.3 Reconstruction Workflow

In general a 3D model is reconstructed in two steps. First, a sparse model (see Section

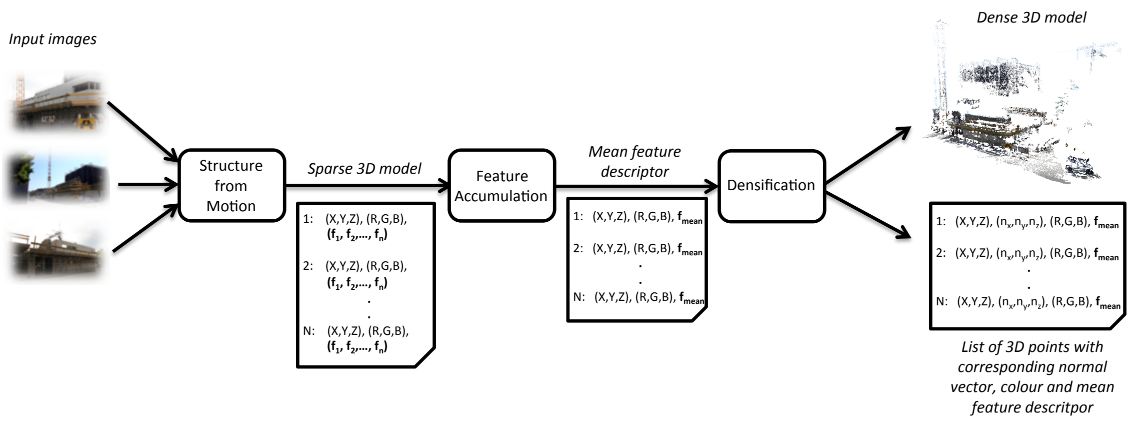

Input images
Structure from Motion
Sparse 3D model
1: (X,Y,Z), (R,G,B), $(\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n)$
2: (X,Y,Z), (R,G,B), $(\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n)$
.
.
N: (X,Y,Z), (R,G,B), $(\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n)$
Feature Accumulation
Mean feature descriptor
1: (X,Y,Z), (R,G,B), $\mathbf{f}_{mean}$
2: (X,Y,Z), (R,G,B), $\mathbf{f}_{mean}$
.
.
N: (X,Y,Z), (R,G,B), $\mathbf{f}_{mean}$
Densification
Dense 3D model
1: (X,Y,Z), (n_x,n_y,n_z), (R,G,B), $\mathbf{f}_{mean}$
2: (X,Y,Z), (n_x,n_y,n_z), (R,G,B), $\mathbf{f}_{mean}$
.
.
N: (X,Y,Z), (n_x,n_y,n_z), (R,G,B), $\mathbf{f}_{mean}$
List of 3D points with corresponding normal vector, colour and mean feature descritpor

Figure 2.14: Flow chart of our 3D reconstruction pipeline. First, a sparse 3D model is reconstructed which is refined through a densification process. This workflow outputs a list of 3D points, where each single point has a 3D position, a colour value, a mean feature descriptor and a normal vector.

2.3.1) is reconstructed which is normally refined by a densification process (see Section 2.3.2). Figure 2.14 shows a flow chart of our 3D reconstruction pipeline. First, a sparse 3D model from a number of images is reconstructed. Figure 2.15 shows such a sparse model.

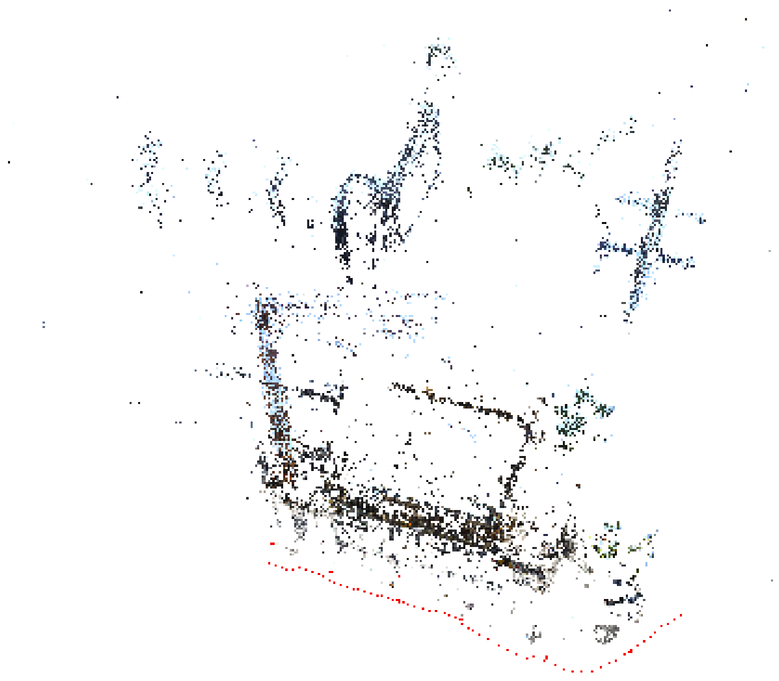
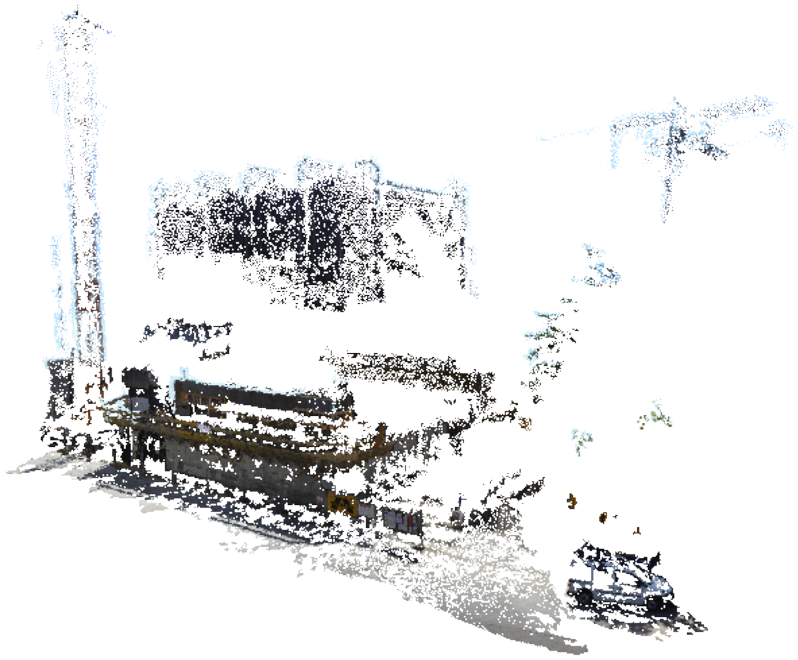


Figure 2.15: Illustration of all cameras, represented in this figure as red dots, which have been localised during the reconstruction process.

Cameras, which have been localised during the reconstruction process, are represented as red dots in this figure. The output of the SfM module is a list of 3D points. Additionally, each point holds colour information and a list of feature descriptors. In the next processing step all features for one point are averaged resulting in one feature descriptor for each 3D point. Finally, a dense model is computed, which gives an additional normal vector for each 3D point.
The difference between a sparse and a dense 3D model is shown in Figure 2.16. Figure 2.16a illustrates a sparse reconstruction of a construction site, while a dense model is shown in Figure 2.16b. The sparse point cloud was reconstructed with Bundler [94, 95]. In addition, for reconstructing the dense model a PMVS / CMVS approach, as addressed in this chapter, was applied.

(a)



(b)

Figure 2.16: Comparison between a sparse and a dense point cloud. a) Illustration of a sparse point cloud reconstructed with the Photo tourism approach. b) Illustration of a dense point cloud applying PMVS/CMVS on the sparse point cloud computed in a).

Chapter 3

Camera Calibration and Video Registration

Contents

The previous chapter explained the 3D reconstruction process in detail. This chapter goes on to introduce the methods and techniques used to register single video frames as well as ordinary images onto the pre-computed 3D model. First, an introduction to camera calibration, which can be loosely classified into photogrammetric calibration and self-calibration methods, is given. Next, the correspondence problem, which is solved through feature descriptors and 2D to 3D matching, is addressed. Subsequently, image distortion models are introduced, as some kind of distortion is present in customary digital cameras. Since the radial distortion model is one of the most widely used, it is explained in more detail including a non-linear optimisation approach which is required to estimate the radial distortion parameters. For robust localisation the DLT algorithm is incorporated into the well-known RANSAC algorithm, since outliers are usually present. Finally, the localisation process utilises some other basic methods, such as bucketing, which are addressed at the end of this chapter.

3.1 Calibration Methods

For the purpose of 3D reconstruction the estimation of camera parameters is an essential factor since it is required for Euclidian reconstruction. These camera parameters are also required for camera localisation. Zhang [123] classified the calibration methods loosely into two major categories, namely *photogrammetric calibration* and *auto-calibration*, respectively.
The photogrammetric calibration method requires the capturing of a number of images of a calibration target. The exact position and orientation of the calibration target in the 3D space must be known. Furthermore, reconstructed 3D models can be used as a reference target as well. In contrast, the auto-calibration methods estimate the camera parameters directly from several uncalibrated images. In general, these methods do not require any calibration target. The camera parameters are estimated by moving the camera within a rigid scene. Moreover, the auto-calibration method can be classified into two specialisations. Typical specialisations are planar auto-calibration, which requires, contrary to standard auto-calibration methods, a calibration target, and methods which initialise the calibration parameters with some prior information. Such initialisation constraints can be selected from the Exchangeable Image File Format (Exif) information of the images.
In this master's thesis we focus on photogrammetric techniques; however, a brief introduction to the auto-calibration methods will be given as well. A comprehensive discussion of calibration methods including both major methods is given in [18].

3.1.1 Photogrammetric Calibration

More than three decades ago, Brown [12] a lot of research had been done in the field of photogrammetric calibration. Several years later, Tsai's [107] very popular and well-known photogrammetric camera calibration had evolved. Tsai's method is a two-step process and additionally incorporates a radial distortion model. In the first step, the orientation and the x and y position of the camera referring to the 3D object reference coordinate system is determined. In the second step, the focal length of the camera, the translation along the z-axis and the radial distortion parameters are estimated.
For calibration purposes a pattern similar to a chessboard is most often used, since corners can be easily extracted with standard corner detection algorithms such as Harris corners [37]. Figure 3.1 shows a typical 3D calibration target, which provides the essential 3D information. In the case of photogrammetric calibration a target generally comprises two or three orthogonal planes. Another possibility to provide such accurate 3D information is to use a reconstructed 3D model. In this thesis a 3D model, as the one computed in the previous chapter, is utilised as 3D calibration target.
Later on, another classic and nowadays widely used approach, known as DLT algorithm, evolved. It was first proposed by Hartley and Zisserman [39] and tries to estimate the camera parameters through a linear transformation. This approach simultaneously estimates

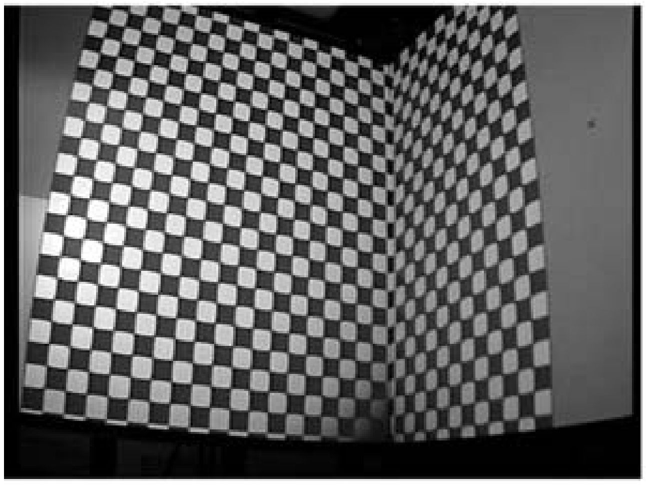


Figure 3.1: Calibration object applied to generate training examples for Tsai's model. This image is taken from [97].

the intrinsic and extrinsic camera parameters. Usually six correspondences between 3D model points and 2D image points are necessary to uniquely estimate the camera parameters, since the camera matrix has 11 DOF and one correspondence solves two equations. As the DLT is used for localisation of the video frames throughout this master's thesis a comprehensive explanation and a motivation for choosing this approach is given separately in Section 3.4.1.

3.1.2 Auto-Calibration

The second category, known as self-calibration or auto-calibration, requires only a number of different images from an arbitrary scene and automatically determines the intrinsic camera parameters from these images. Since no calibration targets with a well-known shape are required for auto-calibration, these approaches can be easily adjusted for 3D reconstruction. Most of the SfM approaches have such an auto-calibration method included, such as Bundler [94] to mention just one. A major advantage of the auto-calibration technique is the fact that no information about the camera itself is required and any standard camera can be used.
Several classifications of auto-calibration techniques are presented in the literature. Azizi [5] classifies the auto-calibration methods mainly into Kruppa equations [23], the absolute quadratic [102] and the modulus constraint [79], as these techniques are called.
Auto-calibration methods can be traced back nearly two decades, when Maybank and Faugeras [66] proposed the theory behind auto-calibration of a moving camera by exploiting the epipolar constraint. Later, Hartley [38] published his work focusing on auto-calibration from multiple views for a rotation camera only. This approach requires at least three images with a different orientation from the same viewpoint. The work

differs fundamentally from the work by Maybank and Faugeras [66], since no epipolar constraints are considered in this approach, because all images are captured from the same point in space. A drawback of this approach is that it requires some kind of pre-calibrated camera parameters. Triggs [103] respectively Mendonca and Cipolla [67] extended Hartley's work. While Mendonca and Cipolla exploit some properties of the essential matrix for stable estimating of varying focal length and principal point, Triggs restricts his approach to planar scenes.

Auto-Calibration Utilising Prior Information. Special cases of auto-calibration methods are techniques which initialise the calibration parameters with some prior information. The Photo tourism approach [2, 94], which is one of the well explored SfM techniques, utilises the Exif information of the images to initialise the intrinsic camera parameters. This Exif information provides a rough estimate of the calibration parameters.

Planar Auto-Calibration. A special case of auto-calibration is the planar auto-calibration technique. This method utilises targets with a chessboard pattern as the photogrammetric calibration methods do. However, in comparison to the photogrammetric calibration methods only planar targets are utilised, as shown in Figure 3.2. One of the well known planar auto-calibration approaches is Zhang's method [123]. Zhang's work aims to calibrate cameras by freely moving a planar pattern or the camera

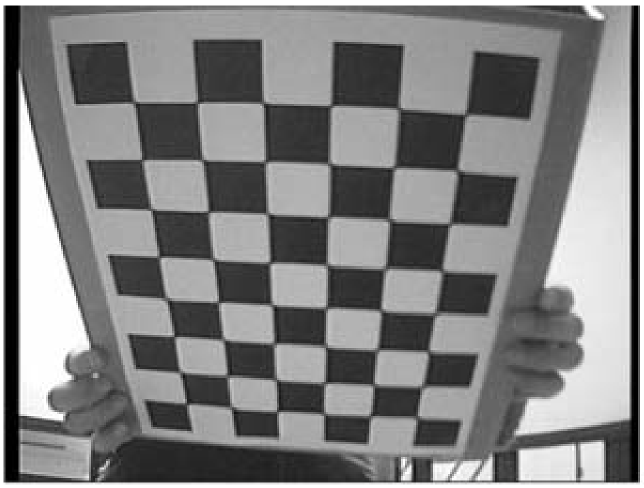


Figure 3.2: Planar calibration target applied to generate training examples for Zhang's model. This image is taken from [97].

itself in a 3D space without any special knowledge of the motion of the object or the camera.
A comprehensive evaluation of Tsai's method, which is a photogrammetric calibration

technique, and Zhang's method in comparison to Heikkila's approach [40] is given in [97]. Another planar auto-calibration method is the approach proposed by Triggs [103].

Auto-Calibration with Printed Distinguishable Markers. Furthermore, Irschara et al. [45] describe a reliable calibration technique which aims to estimate the calibration parameters with the accuracy of a target calibration technique that does not require precise calibration patterns. The method is based on printed distinguishable markers which are deployed in an arbitrary fashion. Moreover, this approach includes a standard radial distortion model as well.

3.2 Correspondence Problem

The correspondence problem, which is ill-posed, describes the problem of finding related points in a pair of images or in multiple images. Solving the correspondence problem can be divided into several parts [39]. The first step is to find similar features across multiple images. This is done by finding the location of a key feature, then describing this key feature through a discriminative feature descriptor and finally matching the computed feature descriptors with other feature descriptors across several images. Different techniques, such as brute force matching or kd-tree nearest neighbour search, can be used to find related points. How to find discriminative features and how to match these features efficiently are explained in the following sections.
Additionally, it should be mentioned that if features are matched across video frames, the method of finding related feature descriptors is known as feature tracking in contrast to feature matching in the case of images [39].

3.2.1 Feature Descriptor

Feature detection, which is a low-level image processing operation, is one of the basic methods in computer vision, since it is fundamental for many applications such as image localisation, object recognition, image stitching, 3D reconstruction and calibration of cameras. It is usually one of the first steps of a processing pipeline, therefore the performance and accuracy of the subsequently applied algorithms strongly depend on the choice of the right feature descriptor. Basically, each pixel is examined and when some requirements are satisfied this pixel is treated as a feature point. Therefore, feature detection results in a set of feature points that are later used to extract the discriminative feature descriptors. In general, three different types of image features are separated.

- Edges
- Corners + Interest points
- Blobs

Canny and Sobel are one of the well-known edge detection approaches, while Harris corners and level curves are typical interest point methods. Furthermore, Laplacian of Gaussian (LoG) and Difference of Gaussian (DoG) can be used to detect interest points as well as blobs [65].
As a consequence of the fundamental importance of feature descriptors, immersive research has been done in the last few decades. Recently, Calonder et al. [15] proposed what is known as the binary robust independent elementary features (BRIEF) descriptor which is set up on the principle of binary strings, hence the matching process is speeded up. Other state-of-the-art feature descriptors are speeded up robust feature (SURF) [6], local energy based shape histogram (LESH) [41], gradient location and orientation histogram (GLOH) [68] and Scale-invariant feature transform (SIFT) [60, 61] to mention only a few. A comprehensive performance evaluation of modern feature descriptors can be found in [68].
One of the most widely used feature descriptors is the SIFT descriptor published by Lowe [61], as it is invariant to scale and rotation which is required for many computer vision tasks. Owing to our modularity implementation, any one of these feature descriptors could be selected for 3D reconstruction and video registration, but we decided to use the SIFT descriptor, because this approach is scale and rotation invariant and is also robust to noise and changes in illumination. Furthermore, the SIFT features are invariant to viewpoint changes and are highly discriminative which is important for our task. Additionally, several open source applications for this approach are available free of charge [10, 110, 115]. Since the SIFT descriptor plays an important role in this master's thesis, detailed information is given in this section.

3.2.1.1 SIFT - Scale-invariant Feature Transform

Since SIFT is utilised as a feature descriptor in this thesis, a brief introduction is given. First, the SIFT algorithm [61] tries to find orientation and scale invariant key features

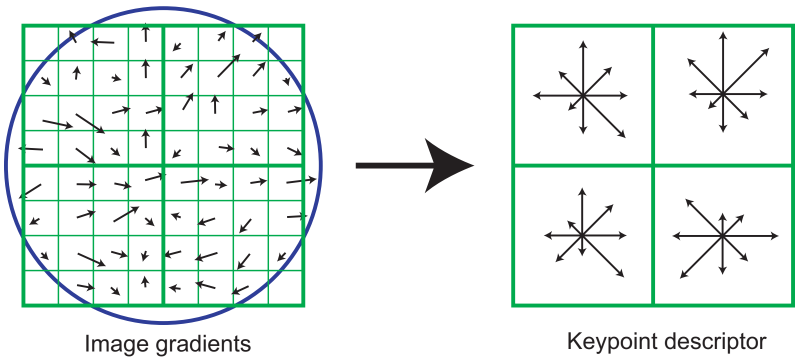


Figure 3.3: Illustration of the principle of the SIFT descriptor for one subregion. This image is taken from [61].

by searching through different scales and positions applying a DoG function. Next, an orientation is assigned to each feature point to remove the rotation dependency. Finally, the feature descriptor can be computed based on the previously found feature points. The basic principle of the creation of such SIFT keypoint descriptors is shown in Figure 3.3. To begin with, the image gradient for each pixel is precomputed and weighted with a Gaussian function (illustrated as the blue circle). For each 4-by-4 subregion the weighted magnitude is accumulated into a histogram with eight bins, as shown in the right part of the figure. This illustration is simplified, since it shows the principle with a patch size of 8-by-8, while throughout this master's thesis a patch size of 16-by-16 is used. This 16-by-16 patch is then divided into eight 4-by-4 descriptors, whereas each descriptor consists of a histogram with eight bins resulting in a 128 dimensional feature descriptor.
Figure 3.4 shows an image from a construction site that is taken from our self-captured datasets and overlaid with the SIFT feature descriptor illustrated as white arrows in the figure. These arrows indicate the orientation and the position of the key features. The

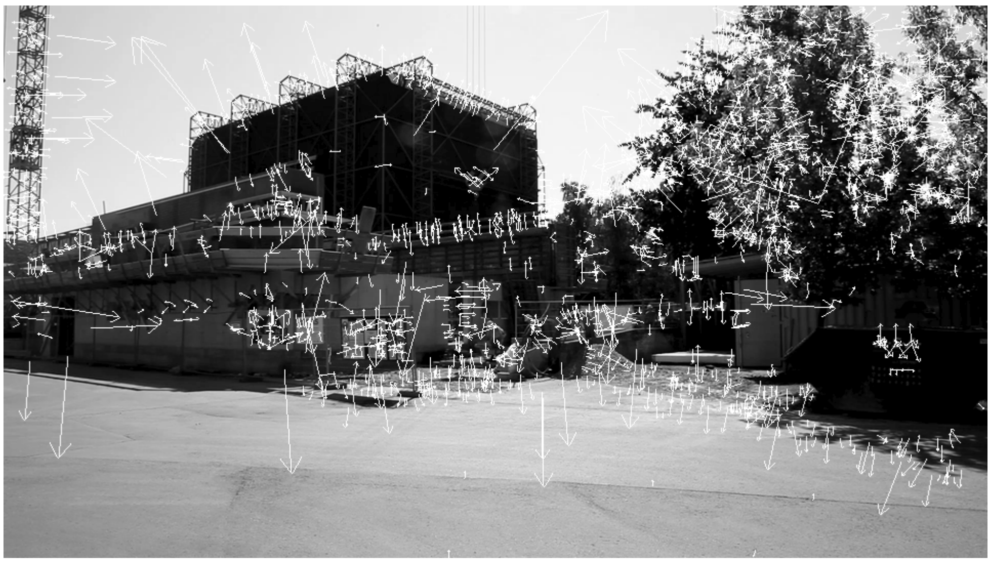


Figure 3.4: Image of a construction site overlaid with computed SIFT descriptors. The arrows show the orientation and the position of the key features.

amount of detected key features depends on the resolution of the underlying image, as addressed by Lowe [61].

3.2.2 2D to 3D Matching

Once the feature descriptors are extracted from the video frames, or from images, the next step is to match these features against the features from the pre-reconstructed 3D model. The matching process is an important part of the registration process as it has an enormous influence on the run-time and the accuracy of the localisation. Generally, each point from the reconstructed point cloud has a position $X = (x, y, z)$ referring to the world

coordinate system, a normal vector $\mathbf{n} = (n_x, n_y, n_z)$, an RGB colour and, additionally, a descriptor. While the position X, the RGB colour and the descriptor are computed during the sparse reconstruction, the normal vector $\mathbf{n}$ is an output of the densification process. The literature proposes various approximation and non-approximation matching techniques with different run-times [7, 8, 21, 28, 71, 84]. The easiest, but computationally most inefficient is what is called *brute force* method, since it compares every feature descriptor from the point cloud with every feature descriptor from the video frame. An advantage of this method is that it is simple to implement and always finds the best match. Nevertheless, the run-time of this method depends on the amount of feature descriptors and the dimension of the feature space itself and increases quadratically with the number of features [8].
A more sophisticated approach is to cluster similar features together and perform a search in the restricted space. The fast library for approximate nearest neighbors (FLANN) is a new approach invented by Muja and Lowe [71] and is based on such a clustering principle. On the other hand, one drawback of these approaches is that they search for an approximate nearest neighbour, while a brute force method always finds the best match.

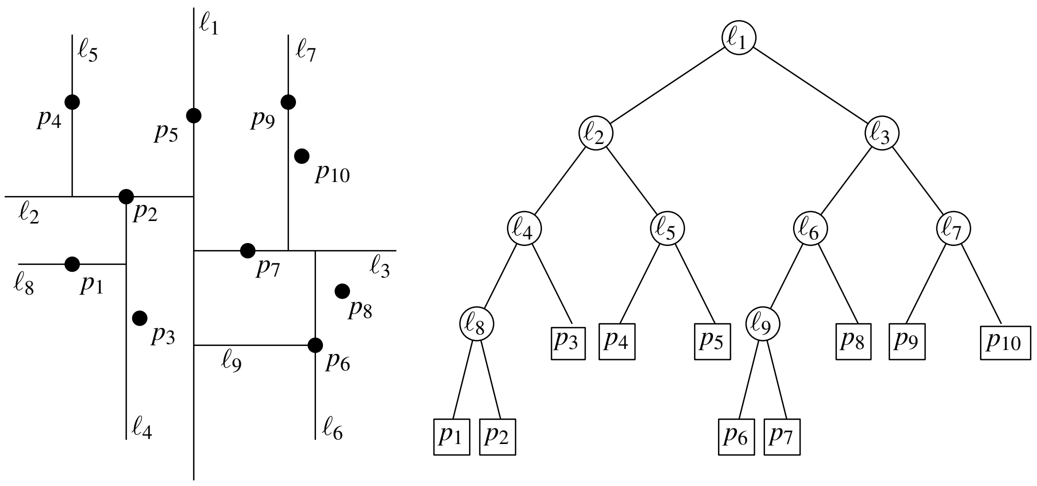
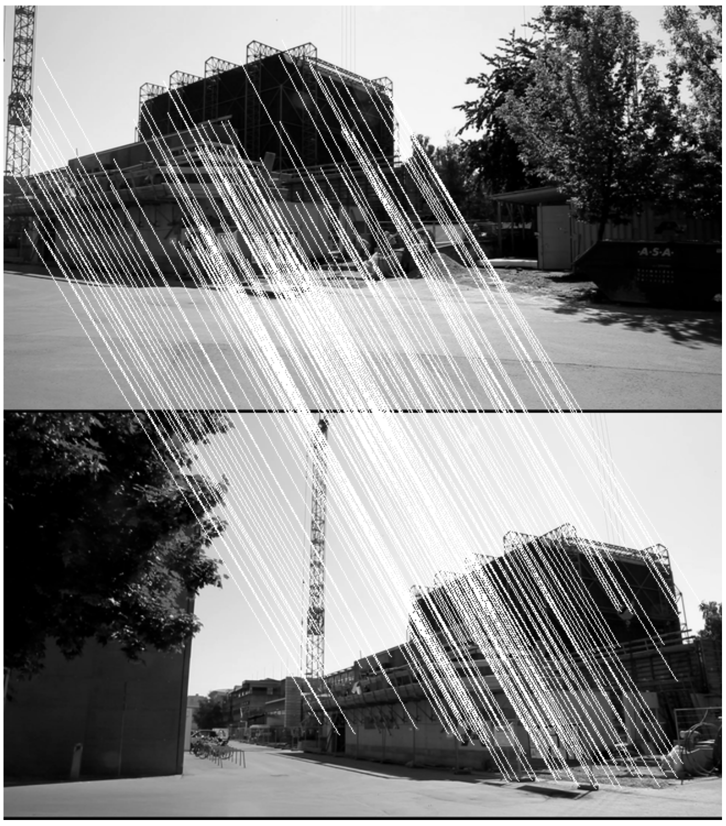


Figure 3.5: Construction principle for a two-dimensional kd-tree taken from [8]. The feature space is divided into subspaces recursively at each level. The left part of this figure shows the subspaces and the corresponding tree nodes are illustrated in the right part of the figure.

The kd-tree principle was first proposed by Friedman [28] in 1977, but since then various enhancements have been published over the years [91]. A kd-tree is built by splitting the entire feature space into subspaces through a hyperplane. Figure 3.5 explains the principle of subdividing for a two-dimensional case. The same principle can be applied for high dimensional cases, as in our case a 128 dimensional feature space. Through this restriction the time for finding a related descriptor can be drastically minimised. As shown by Berg et al. [8] the construction time of the tree for a constant feature space dimension d is $O(n \times logn)$ and the querying time is bounded with $O(n^{1-1/d} + k)$ while a brute force method has a run-time of $O(n^2)$. A detailed explanation, particularly for kd-tree construction, can be found in [8].


(a)

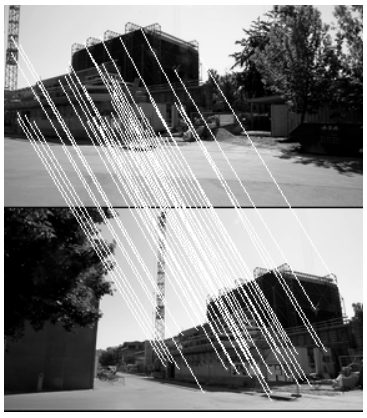

(b)

Figure 3.6: SIFT feature descriptor matching between video frames for two different resolutions. The top images represent the first video frame, while the bottom images were captured three seconds after the first one. a) Image resolution is 653×370. b) Image resolution is 288×163. The white lines connect corresponding feature points. If d_1 is the distance between the query feature and the nearest neighbour and d_2 is the distance between the query feature and the second nearest neighbour, then matches are only accepted if $d_1 < 0.6 \times d_2$ as proposed by Lowe.

The vocabulary tree approach, proposed by Nistler and Stewenius [74], is another efficient matching method, since it scales to large databases and is robust to background cluttering. Since the 2D to 3D matching process is an important bottleneck in the entire localisation process, Sattler et al. [84] proposed a novel approach using direct 2D to 3D matching. While standard direct matching approaches directly search for the nearest neighbour in the feature space, an indirect matching procedure uses an intermediate representation

which does not preserve the proximity between the features. This is especially useful for huge datasets, as the computational cost increases more slowly than for direct matching techniques.
Figure 3.6 shows the matching between two images, taken from our self-captured datasets, using SIFT feature descriptors and a nearest neighbour search for the matching purpose. As shown in both figures most of the extracted feature points are matched correctly, but there are still some false matches; hence, robust methods have to be applied in the next step of the workflow to get accurate results. This is done by adding a RANSAC algorithm in the registration process, as described in Section 3.4.3.

3.3 Image Distortion

Most of the standard cameras have some sort of image distortion included [42]; however, many standard image applications determine that distortion can be neglected. On the contrary, for applications such as metric reconstruction and image-based measurements distortion has a great influence on the accuracy. A comparison of a model including and excluding a radial distortion model is given in Section 5.3.6.
First, in this section some general distortion models are addressed. Next, since the radial component has the greatest influence on the accuracy this model is explained in more detail. Finally, the Levenberg-Marquardt optimisation approach is introduced.

3.3.1 General Distortion Models

Nearly one century ago, in 1919, the first research into camera lens distortion was undertaken, as addressed by Wang's article about a new camera lens distortion model [113].

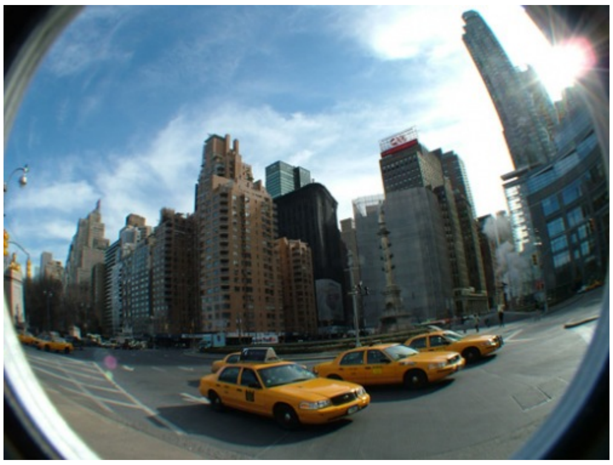


Figure 3.7: New York City seen through a fish-eye lens with a high barrel distortion, which is a type of radial distortion[1].

[1]This image is taken from http://augenklick.wordpress.com

The focus at that time was on decentering distortion, which consists of tangential and radial components. Nowadays, *thin-prism*, *radial* and *tangential* distortion components are the most investigated; however, other components can still be present. Figure 3.7 shows a photograph of New York City through a fish-eye lens, which has a high barrel distortion, which is one type of radial distortion.
The various distortion components can be described with several mathematical models and parameters respectively. Since the accuracy of many applications can be increased through modelling of the distortion parameter a great deal of literature regarding this topic is available. Over the years, several approaches with different assumptions have been published [98, 107, 114], however the fact remains that the main research focus was on radial distortion models, as the immersive literature about this subtopic shows. The polynomial model, which is the simplest one, the rational model and the field of view (FOV) model are a small selection of the most used radial distortion models, alternatively more complex models can be found in the available literature.

3.3.2 Radial Distortion

As addressed in Section 3.3 the radial distortion model is the best investigated one and for most applications introducing such a model is sufficient. It can be classified into *barrel* and *pincushion* distortion [119], as shown in Figure 3.8.

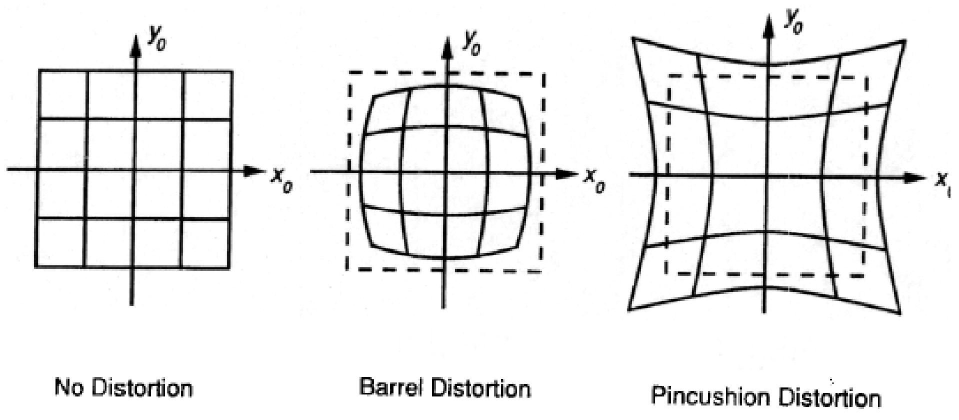


Figure 3.8: Illustration of the two different radial distortion models[2].

For barrel distortion the image magnification decreases with increasing distance from the principal axis. In contrast, for pincushion distortion, the image magnification increases with increasing distance from the principal axis. A typical example of barrel distortion is the fish-eye lens, while binoculars show pincushion distortion effects. Furthermore, a combination of both radial distortion components, known as *mustache* distortion, can be present as well [32].

[2]This image is taken from http://www.uni-koeln.de

Next, we are going to introduce a mathematical model, like the one proposed by Hartley and Zisserman [39], to compensate for the radial distortion

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = L(\tilde{r}) \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}, \tag{3.1}$$

where $(\tilde{x}, \tilde{y})$ represents the ideal image position and (x_d, y_d) is the position after applying the radial distortion model. The term $L(\tilde{r})$ is a multiplication factor depending only on the radius $\tilde{r}$, which is computed by

$$\tilde{r} = \sqrt{(\tilde{x} - x_c)^2 + (\tilde{y} - y_c)^2}, \tag{3.2}$$

where (x_c, y_c) represents the radial distortion centre. Various more, or less, complex models to describe the radial distortion factor $L(\tilde{r})$ are addressed in the literature. In general, it can be expressed by a Taylor expansion, such as

$$L(r) = 1 + k_1 r + k_2 r^2 + k_3 r^3 + ... + k_n r^n, \tag{3.3}$$

as proposed by Hartley and Zisserman. Since higher terms of the Taylor approximation can be neglected the distortion factor is simply computed by

$$L(r) = 1 + k_1 r^2 + k_2 r^4, \tag{3.4}$$

where k_1 and k_2 represent the two radial distortion parameters, which are estimated through a non-linear estimation (see Section 3.3.3). Finally, these two parameters together with the radial distortion centre (x_c, y_c) can be seen as additional interior camera parameters

$$D = [k_1, k_2, x_c, y_c]^T. \tag{3.5}$$

If we assume the distortion parameters are known the image can be undistorted by applying

$$\hat{x} = x_c + L(r)(x - x_c) \tag{3.6}$$

and

$$\hat{y} = y_c + L(r)(y - y_c), \tag{3.7}$$

where $(\hat{x}, \hat{y})$ are the undistorted pixels and (x, y) represent the distorted pixels. The radius r is computed as in Equation (3.2) and the distortion function as in Equation (3.4).
An alternative and more simple model, in comparison to the radial distortion model proposed by Hartley and Zisserman, is the division model published by Fitzgibbon [27], which

only depends on one parameter k and is defined as

$$\hat{x} \sim \frac{x}{1 + kr_d^2} \tag{3.8}$$

and

$$\hat{y} \sim \frac{y}{1 + kr_d^2}. \tag{3.9}$$

The work of Bujnak et al.[14] is built upon Fitzgibbon's model, however the major innovation of this approach is to include the estimation of the radial distortion parameters directly into the RANSAC loop. Since the distortion parameters are estimated for each consensus set in the RANSAC step the outcome is more accurate. As a consequence, we decided to apply the same approach to a more complicated radial distortion model. To be precise, we used the distortion model proposed by Hartley and Zisserman. In contrast to Fitzgibbon's one parameter model, this model has four parameters; therefore, a more accurate outcome can be expected for a slightly higher overhead in the optimisation process. Furthermore, the radial distortion model applied in this thesis is the same as the one used by Irschara et al. [45].

3.3.3 Non-linear Optimisation

As shown in Section 3.3.2 the distortion model is generally represented by several distortion parameters, such as k_1, k_2,…,k_n. These distortion parameters can be estimated through a non-linear optimisation process, which aims to minimise an objective function. Several iterative and calculus based (non-iterative) methods are proposed in the literature. Newton invented the first iterative optimisation method, which was later refined by Gauss, while Largrange invented the first closed form solution.

One of the most efficient and well-known curve fitting methods is the Levenberg–Marquardt algorithm (LMA) invented by Levenberg [54] in 1944 and refined by Marquardt [64] two decades later. The LMA method utilises both the Gauss-Newton algorithm and the gradient descent technique to estimate the parameters of the non-linear objective function. In comparison to the Gauss-Newton algorithm the LMA is more robust, but for some configurations slower than the Gauss-Newton method and only a local minimum can be found. In the case of multiple minima the initial guess has to be close to the global minima, otherwise the global minima cannot be found, since a non-linear optimisation method only finds a local minima. For convex functions, which only have one global minima, the initial guess is unimportant, since the LMA always finds a minimum. The LMA works in an iterative way and tries to minimise Equation (3.10) in every iteration until a pre-defined bound ϵ is reached. [54, 64]

In general, for a non-linear function $F : \mathbb{R}^m \to \mathbb{R}^n$, $m < n$ the LMA method tries to minimise

$$S(\mathbf{p}) = \sum_{i=1}^{n} [y_i - f(x_i, \mathbf{p})]^2, \tag{3.10}$$

where x_i is the input of the objective function and y_i is the output. Like the Gauss-Newton method the LMA substitutes, in each iteration, the non-linear function through a linearisation and tries to solve this linearisation problem

$$\min_{x \in \mathbb{R}^m} \|F(x_k) + J(x_k)(x - x_k)\|_2^2, \tag{3.11}$$

where J is the Jacobi matrix of the function f. Through an additional constraint $\|x - x_k\|_2^2 < r_k$ the error is minimised in each iteration. The outputs of this algorithm are the optimised parameters $\mathbf{p} = (p_1, p_2, \ldots, p_k)$.

Basically, our novel approach incorporates two steps, which are applied consecutively in one iteration. In other words, from the pre-computed 2D to 3D correspondences an initial camera pose is estimated and refined with the Levenberg Marquardt optimiser. Whilst one set of parameters are fixed the other parameters are refined and vice versa. More specifically, in the first step the camera pose and the intrinsic camera parameters are estimated through a standard 6-point DLT algorithm as addressed in Section 3.4.1. Furthermore, in the second step the previous estimated camera pose is fixed and the distortion model is refined. This refined distortion model leads to a more accurate camera pose in the next iteration, since the re-projection error is minimised through the undistortion of the image. This procedure is repeated until the radial distortion parameters convergence. Since many outliers are present in a standard set of 2D to 3D correspondences this two-step algorithm is included into a RANSAC loop, whereas the radial distortion is only refined if the consensus set contains a minimum number of inliers. The number of RANSAC iterations depends on several parameters, such as percentage of outliers, confidence of the model and the empirical threshold parameters, as described in Section 3.4.3. Finally, after a final refining of the camera pose with all inliers detected through the RANSAC iteration the radial distortion is refined as well, starting the optimisation process from the precomputed initial values. This refinement step includes the entire set of inliers found during the RANSAC iteration, while the previous optimisations only use the actual consensus set.

3.4 Camera Pose Estimation

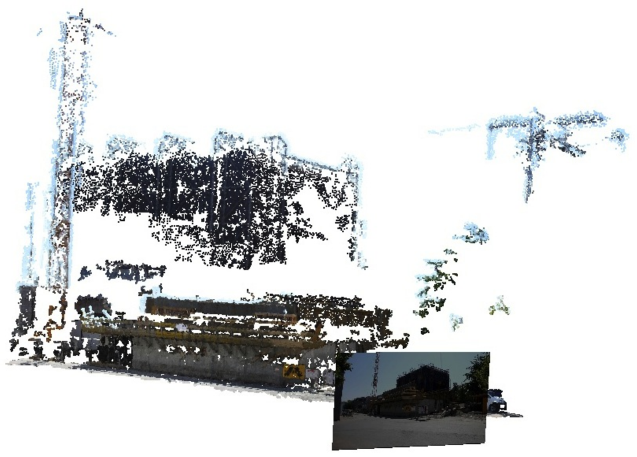
Camera pose estimation, also known as image localisation or image registration, is essential for many computer vision applications, especially in the field of 3D reconstruction. Immersive research has been done in the last few years, hence an extensive amount of literature regarding image localisation is available [92].
The camera pose estimation approaches can be classified as to the number of known correspondences. The most popular methods are 3-point [36], 4-point [104], 5-point [104] and 6-point [39, 106] algorithms. A 3-point algorithm requires three known correspondences, while a 6-point algorithm needs a minimum of six correspondences.

Recently, several image-based localisation applications have evolved. Since GPS is restricted to a line of sight to the sky and is not applicable in indoor environments, image-based localisation provides a serious alternative. Another advantage of this technique is its high accuracy. For example, finding the position of a mobile device can be interesting for many applications, such as guiding a tourist. Other applications are navigation of robots or pedestrians and augmented reality, as mentioned by Sattler et al. [84].
As camera pose estimation is a highly relevant research topic and registration of video frames is a major part of this master's thesis, some related work is addressed in the following paragraph. Robertson et al. [82] match images, which are captured with a mobile device, against a database of views to find the position of the mobile device. Zhang's [121] approach is similar but focuses on urban environments. Schindler et al. [87] proposed a novel location recognition method introducing a vocabulary tree. Simultaneous object localisation and camera estimation on a pre-computed 3D model is explained in [117]. Irschara et al. [46] proposed a novel method for fast location recognition by registering images on to a sparse 3D model. Recently, Sattler et al. [84] demonstrated that direct 2D to 3D matching techniques based on a quantised visual vocabulary tree and a DLT algorithm included into a RANSAC framework greatly improves the performance of image registration. Another work assuming the 3D model as known, utilises the 6-point DLT algorithm to register the images [118]. A similar method using a modified DLT algorithm is addressed by Triggs [104]. Kaminsky et al. [51] try to align SfM triangulated points with overhead images, like floor plans and satellite images, by matching the 3D point cloud to the edges of the images. Another more general approach is demonstrated by Li et al. [56], which uses adaptive, prioritised SIFT descriptors to speed up the time intensive matching process. The previously addressed projects are only a select few of many, various other interesting projects can be found in the literature.
Figure 3.9 shows the registration of three video frames to a pre-computed 3D triangulated model reconstructed with a state-of-the-art SfM method after applying Algorithm 1 from Section 3.4.1. The images and videos are taken from our self-captured datasets.

3.4.1 6-Point Direct Linear Transform

The DLT is a popular method in the field of projective geometry, as it can estimate the projection matrix P from a set of given 3D scene points and the corresponding 2D image points. It was first proposed by Hartley and Zisserman [39]. Trucco and Verri also provide a detailed explanation regarding the DLT algorithm [106]. The 6-point DLT algorithm makes no assumptions about the intrinsic and extrinsic camera parameters; therefore it is applicable for any configuration without further knowledge about the camera type or geometry. In general, the DLT method tries to solve

$$\mathbf{x}_k \propto \mathbf{A}\,\mathbf{y}_k \tag{3.12}$$


(a)

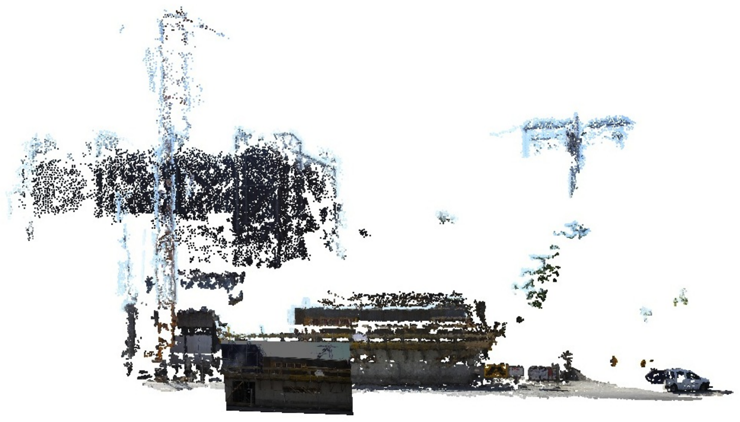

(b)

Figure 3.9: Video frame registration for various viewpoints after applying Algorithm 1 to a pre-computed 3D model.

for $k = 1, \ldots, N$. $\mathbf{x}_k$ and $\mathbf{y}_k$ are given vectors and $\mathbf{A}$ denotes a matrix including the unknown parameters. Originally the DLT was proposed to find the homography between related 2D image points by taking $n \geq 4$ point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. It determines the unknown parameters of the 3×3 matrix H of the homogenous transformation

$$\mathbf{x}_i = H\mathbf{x}'_i. \tag{3.13}$$

The DLT algorithm [39] is widely used in projective geometry; though, there are now several modifications of the original DLT algorithm [84, 104]. The procedure can be easily modified so it can be applied to camera pose estimation and image registration respectively. As this algorithm is used for camera pose estimation and video registration throughout this master's thesis, a comprehensive explanation is given in the next paragraph [39].

Given $n \geq 6$ correspondences between image points and scene points $\mathbf{x}_i \leftrightarrow \mathbf{X}'_i$ the algorithm determines the 3×4 camera projection matrix P. The homogenous transformation is given by

$$\mathbf{x}_i = P\mathbf{X}'_i \tag{3.14}$$

for all correspondences i. For each relation $\mathbf{x}_i \leftrightarrow \mathbf{X}'_i$ a correspondence can be derived

$$\begin{pmatrix} \mathbf{0}^T & -w_i\mathbf{X}_i^T & y_i\mathbf{X}_i^T \\ w_i\mathbf{X}_i^T & \mathbf{0}^T & -x_i\mathbf{X}_i^T \\ -y_i\mathbf{X}_i^T & x_i\mathbf{X}_i^T & \mathbf{0}^T \end{pmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \\ \mathbf{P}^3 \end{pmatrix} = \mathbf{0}, \tag{3.15}$$

where each $\mathbf{P}^{iT}$ is a vector of size 4x1, containing the i-th row of the projection matrix P. The three equations resulting from Equation (3.15) are not linearly independent, therefore the first two equations are enough and as a consequence the previous terms can be simplified to

$$\begin{pmatrix} \mathbf{0}^T & -w_i\mathbf{X}_i^T & y_i\mathbf{X}_i^T \\ w_i\mathbf{X}_i^T & \mathbf{0}^T & -x_i\mathbf{X}_i^T \end{pmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \\ \mathbf{P}^3 \end{pmatrix} = \mathbf{0}. \tag{3.16}$$

Each $\mathbf{x}_i \leftrightarrow \mathbf{X}'_i$ correspondence results in two equations, as shown in Equation (3.16). Therefore, six such relations result in 12 linear independent equations, which are stacked together in a 2n×12 matrix A and subsequently an singular value decomposition (SVD) is applied to solve this linear system. In general, an SVD decomposes a matrix M of size m×n

$$M = UDV^*, \tag{3.17}$$

into a unitary matrix U of size m×m, a diagonal matrix D with size m×n including the eigenvalues and an n×n matrix V. The projection matrix P is then determined by taking the unit singular vector from matrix V which relates to the smallest singular value in matrix D [39].
A summarisation of the entire camera pose estimation is given in Algorithm 1, which is taken in a modified form from Hartley and Zisserman [39].
Normalisation of the input data is an essential part of the entire algorithm. The image points x_i should be normalised in such a way that their centroid is at the origin. Additionally, these input values have to be scaled such that the root mean squared (RMS) distance between origin and the image points is equal to $\sqrt{2}$. These requirements result in the transformation matrix T_1

$$T_1 = \begin{pmatrix} s & 0 & -sc_x \\ 0 & s & -sc_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{3.18}$$

Algorithm 1 Camera Pose Estimation

```
Input:   $n \geq 6$ world point (3D) to image point (2D) relations $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$
Output:  Projection Matrix $P$

ALGORITHM:
(1) Normalisation:
A similarity transform (see Equation (3.18)) $\tilde{\mathbf{x}}_i = T_1\mathbf{x}_i$ is applied for normalisation of the image points $\mathbf{x}_i$
A similarity transform (see Equation (3.19)) $\tilde{\mathbf{X}}_i = T_2\mathbf{X}_i$ s applied for normalisation of the space points $\mathbf{X}_i$

(2) DLT:
A 2n×12 matrix $A$ is formed by putting Equation (3.16) generated from the relations $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{X}}_i$. Through an SVD and by taking the unit singular vector with the smallest related singular value a solution for $A\mathbf{p} = \mathbf{0}$ is obtained.

(3) Denormalistion:
P in the original coordinate system is obtained by de-normalisation with $P = T_1^{-1}\tilde{P}T_2$.
```

where s denotes the scaling factor and (c_x, c_y) represents the centroid, which is determined by taking the mean overall input values x_i. For the scene points the normalisation is similar to the normalisation of the image points. First, the centroid of the scene points is translated to the origin and then, the input values are scaled such that the RMS distance between the origin and the scene points is equal to $\sqrt{3}$. This leads to the transformation matrix T_2

$$T_2 = \begin{pmatrix} s & 0 & 0 & -sc_x \\ 0 & s & 0 & -sc_y \\ 0 & 0 & s & -sc_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3.19}$$

where s represents the scale and (c_x, c_y, c_z) denotes the centroid of all input points X_i.
Another point, which should be mentioned, is the possibility of degenerated configurations. For the determination of the projection matrix P two degenerated configurations are important, which are shown in Figure 3.10. Figure 3.10a shows a degenerated case, where points and camera lie on a twisted cubic. In the second case, shown in Figure 3.10b, the points lie on a single plane and a single line that contains the camera intersects with this plane. For configurations which are close to these degenerated cases the accuracy of the projection matrix P decreases [39].
One simple way to avoid such degenerated cases is to split the entire image into buckets, as they are termed, (also known as binning) and only draw one point from each bucket [124]. This approach is explained in more detail in Section 3.4.4.
Moreover, if just six correspondences are used the algorithm computes an exact solution

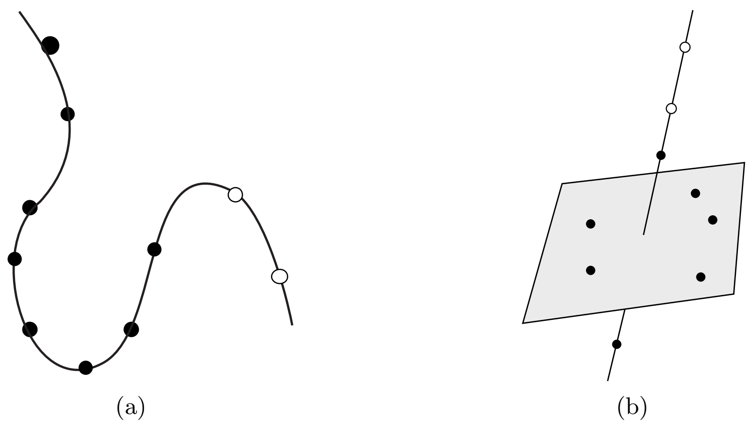


Figure 3.10: Two critical degenerated configurations. The white dots denote the camera centre, while the black dots represent the points. These images are taken from [39].

for the projection matrix P; however, if the points include some noise an over-determined system can be applied. In this case the solution is to minimise a geometric or an algebraic error over all input points [39].
Furthermore, this algorithm can be used for camera calibration. In general, one image is enough to estimate the intrinsic and extrinsic camera parameters; however, robust averaging leads to more constant parameters. If the intrinsic camera parameters are known a 3-point algorithm can be applied instead of the 6-point DLT which, in general, results in more accurate results. Additionally, there is a noticeable speed up in the computation time.

3.4.2 3-Point Algorithm

Section 3.4.1 explained the standard 6-point DLT, which makes no assumptions about the intrinsic and extrinsic camera parameters; therefore the 3×4 projection matrix has eleven DOF (up to an unknown scale) and exactly $5\frac{1}{2}$ correspondences are necessary to solve this linear system. This means that for five points we have to know the correspondence for the x and the y coordinate and for the last point only the correspondence for the x or the y coordinate is required.
Like the 6-point DLT algorithm a 3-point algorithm [36] can be applied to estimate the projection matrix P; however the 3-point algorithm assumes the intrinsic camera parameters as known. As shown in Equation (2.4), the intrinsic camera matrix K has five DOF; therefore the extrinsic camera matrix, containing the camera orientation and the camera position, has to have six DOF. To solve a linear system with six unknowns six linear independent equations are required; hence, three point correspondences are necessary as every point leads to two equations.
Since in many applications the same camera is used to capture the images or videos

throughout one task, it is more efficient to first calibrate the camera with the 6-point DLT as described in Section 3.4.1 and afterwards apply a 3-point algorithm to estimate the extrinsic camera parameters. A comparison between the 6-point DLT algorithm and the 3-point algorithm regarding speed up is given in Chapter 5.

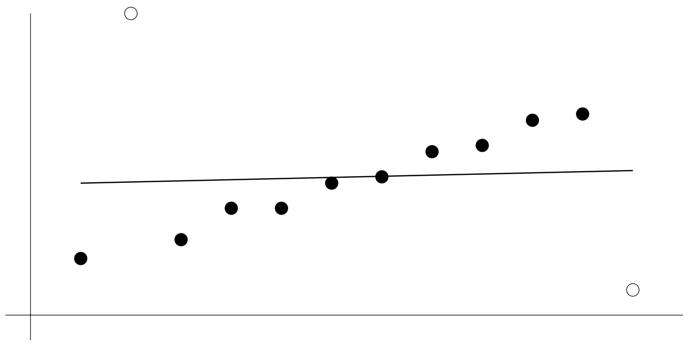
3.4.3 Robust Estimation with RANSAC

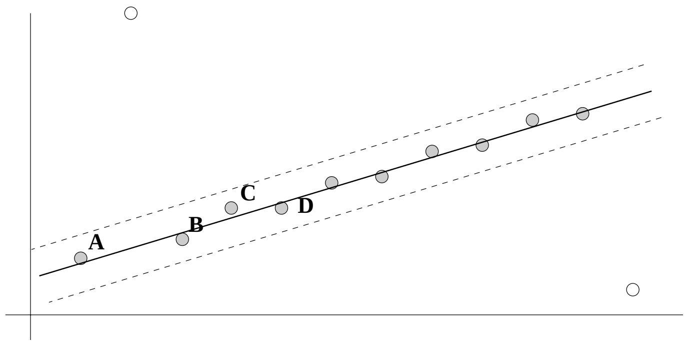
So far, we have assumed that each 3D scene point is matched to the corresponding 2D image point; however through introduced noise in the measurement process and because of features that are not unique this is generally not the case. Therefore, a robust method to detect these false matches is required. These mismatched points, also termed outliers, have a substantial influence on the accuracy of the video frame registration procedure, since a few outliers can lead to a completely different model. The influence of such outliers for a simple line fitting model is shown in Figure 3.11. The model consists of ten inliers (black dots) and two outliers (white dots). Figure 3.11a shows the fitted line after the application of a simple least-square solution including all twelve input points. As it is clear to see, the line does not fit the model very well. Using the RANSAC algorithm instead leads to a much better model as illustrated in Figure 3.11b, since the two outliers are not included in the final model fitting process. To sum up, the RANSAC method seeks to detect such outliers and additionally endeavours to fit a model to the remaining data by minimising the error between the estimated model and the remaining data points. A comprehensive performance evaluation of the RANSAC family is given by Choi et al. [17].
Algorithm 2 is a modified version of both algorithms, the original RANSAC algorithm invented by Fischler and Bolles [25] and the adapted one by Hartley and Zisserman [39]. It is adapted in such a way that it is suitable to estimate the projection matrix P robustly. Some more supplements are provided by Trucco and Verri [106].

Several parameters affect the performance of the RANSAC algorithm:

- n: Defines the minimum number of points required to estimate the model.
- k: Defines the number of iterations of the RANSAC algorithm.
- t: Threshold value defining whether a point fits the pre-computed model. If the projection error of one point is smaller than t it is added to the consensus set.
- d: Threshold value defining the number of input points which have to support the estimated model.

Since the RANSAC performance and the accuracy of the localised frames are highly dependent on these parameters a comprehensive explanation about the choice of these parameters is given in the next paragraph.
As mentioned in Section 3.4.1 the projection matrix has 11 DOF. The RANSAC algorithm


(a)



(b)

Figure 3.11: The RANSAC algorithm aims to fit a line to the given input points by excluding outliers and minimising the error between the fitted model and the inliers. a) Line fitted with a standard least-squared method approach taking all points into account. Black dots denote inliers, while white dots represent the outliers of the model. b) Line fitted after applying the RANSAC method which detects the two outliers and fits a line to the remaining points. These images are taken from [39].

usually takes as input the minimum required points to estimate a model, hence $n = 6$. The number of iterations k is chosen in such a way that with some possibility p a set of inliers is found. If we suppose that w is the probability of finding an inlier set, then $e = 1 - w$ is the probability of finding a set of outliers; therefore we can write

$$(1 - w^n)^k = 1 - p, \tag{3.20}$$

which results in

$$k = \frac{\log(1-p)}{\log(1-(1-e)^n)}. \tag{3.21}$$

In general, p is chosen to be greater than 0.99. Table 3.1 illustrates the required number of iterations k depending on the probability p that at least one consensus set is free of outliers and on the percentage of the outliers itself. For example, for a dataset containing

Algorithm 2 RANSAC algorithm for robust estimation of the projection matrix P

```
while $it < k$ do
    1. Randomly select $n$ points from the entire set $S$ and determine the projection matrix $P$ applying the DLT algorithm (see algorithm 1) to the selected subset $S_i$.
    2. For each point not included in the subset $S_i$ compute the projection error by using the previous computed projection matrix $P$.
    3. Each point which has a projection error smaller than the distance threshold $t$ is added to the set $S_i$. This set is also termed consensus set and defines the inliers of $S$.
    4. If the size of $S_i$ (the number of inliers) is greater than a definable threshold $d$, a good model has been found. Re-estimate (applying DLT) the model using all points in $S_i$ and determine the normalised projection error regarding all elements in the consensus set $S_i$.
    5. If this normalised projection error is smaller than the previous one, a better model has been found and is saved.
    6. The radial distortion parameters are refined.
end while
```

Finally, the saved model with the smallest projection error is returned.

	Outliers								
p	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7
0.9	1	2	4	8	19	49	147	562	3157
0.95	2	3	4	10	24	63	191	730	4107
0.99	2	4	7	16	37	97	293	1122	6314
0.999	3	6	10	23	56	145	439	1683	9472

Table 3.1: Required iterations depending on the probability p that at least one consensus set is free of outliers and on the percentage of the outliers itself. The values are computed applying Equation (3.21) and $n = 6$.

40% of outliers 97 iterations are required to get an inlier set with a probability of 0.99. The parameter t is usually chosen empirically. Since the radial distortion model is not treated in the first iteration in our case, it makes sense to choose a slightly greater value for t at the beginning and decrease it in the following iterations.
A reasonable size of the consensus set d is

$$d = (1 - e)N \tag{3.22}$$

where e is the proportion of outliers and N denotes the total number of points.
In conclusion, the RANSAC algorithm is powerful when datasets with outliers are present, since it can detect such outliers and can fit a robust model by taking only the remaining inliers into account.

3.4.4 Bucketing

As described in Section 3.4.3, the RANSAC algorithm randomly selects n elements from the entire input dataset S. Moreover, as addressed in Section 3.4.1 degenerated config-

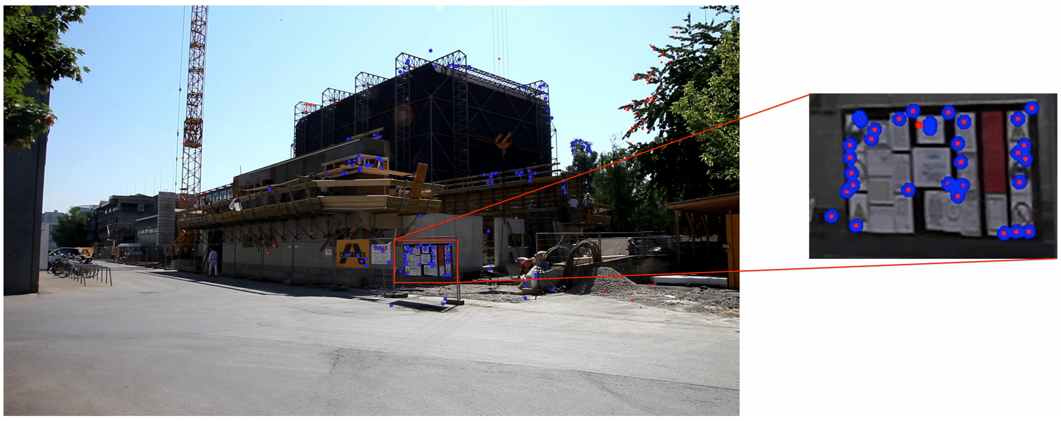


Figure 3.12: Due to the high compression of feature descriptors in several image parts, the estimated camera pose can be extremely inaccurate if several features are selected from the same area of the image.

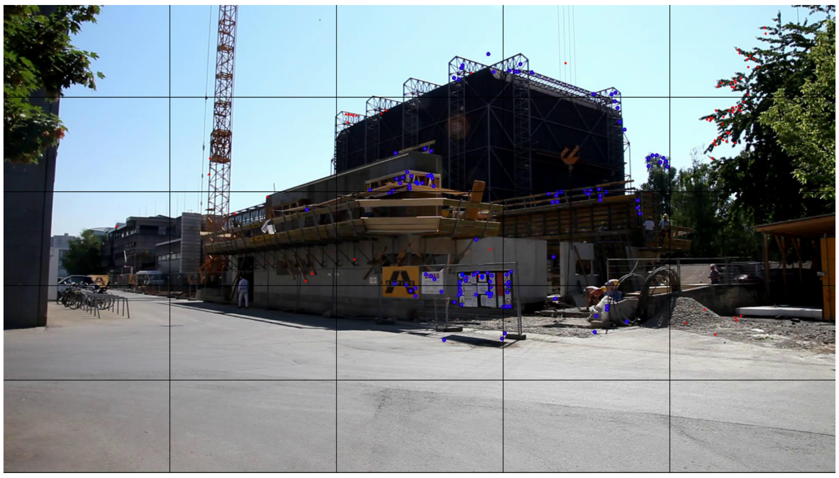


Figure 3.13: Illustration of the bucketing principle for an exemplar grid size of 5x5. First, n grid elements, also termed buckets, are selected randomly. Next, one element from each previously selected bucket is drawn randomly. With such a technique the probability of degenerated cases decreases.

urations can lead to poor results regarding accuracy and it can even lead to a failure of the localisation process. One such degenerated configuration (compare with Figure 3.10b) is shown in Figure 3.12, where the feature points lie on a single plane. Selecting points mainly from this area of the image leads to inaccurate camera poses.
A solution to this problem is to split the input data space S into a subspace [124]. Figure 3.13 shows an image divided into such subspaces, which are also known as bucketing or binning. Empirical evaluations with our datasets have shown that a grid size of 5×5 results in the smallest re-projection error. Instead of randomly selecting n elements from the entire input set, a more sophisticated technique is to first randomly select n mutual subspaces and in the next step choose one point randomly from each subspace. This technique is called bucketing and leads to selected samples which are spread over the entire image. It was first proposed by Zhang et al. [124] in a way slightly different to our approach.

3.4.5 Localisation Workflow

This section briefly discusses our localisation approach, which is shown in Figure 3.14. First of all, features are extracted from the video frames. Next, these features are matched

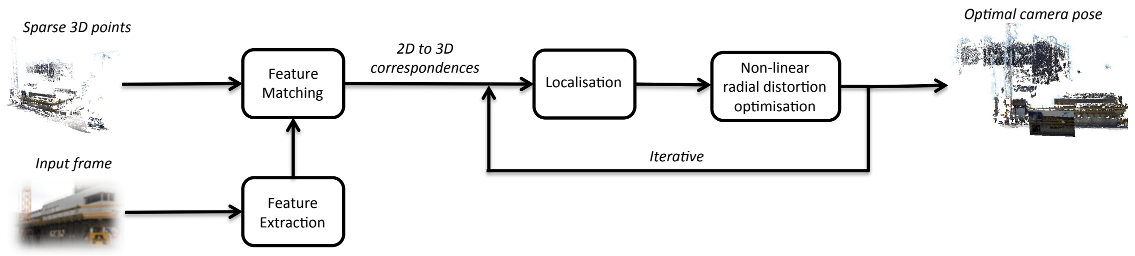


Figure 3.14: Overview of the localisation workflow. The camera pose is estimated in an iterative way utilising a RANSAC framework and a non-linear optimiser.

against the sparse 3D point cloud resulting in a set of 2D to 3D point correspondences. These correspondences are required for the camera localisation. The localisation process works in an iterative way. First, the camera position is determined through a 6-point algorithm included into a RANSAC framework. Subsequently, the radial distortion parameters are estimated through a non-linear optimisation, which minimises the re-projection error. These two steps are repeated until the model converges. The final outcome is an estimated camera pose with corresponding radial distortion parameters.

3.5 Summary

Several improvements have been made in this thesis. First of all, directly including the estimation of the radial distortion parameters into the RANSAC loop significantly minimised the re-projection error. Furthermore, adding a refinement step after finding the best hypotheses improved our outcomes immensely. Additionally, restricting the 2D to 3D to the 200 best matches resulted in better localisation results and enormous speed-up. Moreover, estimating four parameters for the radial distortion model instead of only two, as most approaches do, led to more accurate results. Finally, the bucketing concept minimised the number of unregistrable frames, since degenerated cases are avoided.

Chapter 4

3D Video Rendering

Contents

In the previous chapters we addressed the 3D modelling process, the camera calibration process and the video localisation process in detail. All these steps are finally required to render a realistic 3D video from a given 2D video. The 3D video generation process adds an additional dimension to the 2D information supplied, known as *depth information.* This chapter addresses the estimation of these depth values in detail. First of all, the depth value estimation process requires some kind of consistent regions, which can be determined through unsupervised segmentation algorithms. The appearance of the 3D video depends heavily on this segmentation algorithm; hence, it is addressed comprehensively in this chapter. As already mentioned, the depth information can be used for several other applications as well; therefore, the generation of depth maps and an approach to detect changes between the 3D model and the registered 2D video are addressed.
Finally, this chapter presents a flowchart of the entire workflow, which gives an overview of the various parts of the video registration pipeline and specifically shows the interaction between the different modules.

4.1 3D Video Rendering

In the previous two chapters the 3D modelling process (see Chapter 2) and the video localisation process including the estimation of the intrinsic camera parameters (see Chapter 3) were comprehensively discussed. Now, this section combines all these pre-computation steps and renders a realistic 3D video from a given 2D video sequence through estimating a depth value for each superpixel in the video. Figure 4.1 illustrates the workflow of 3D video rendering process.

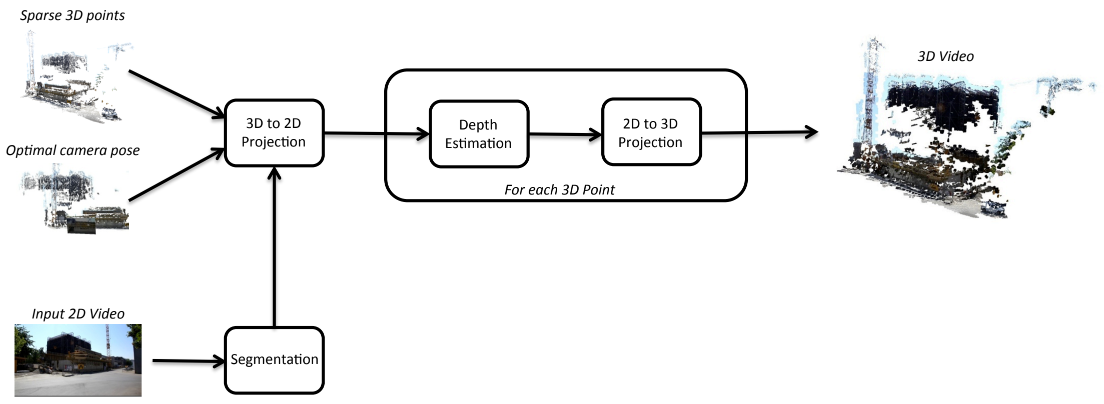
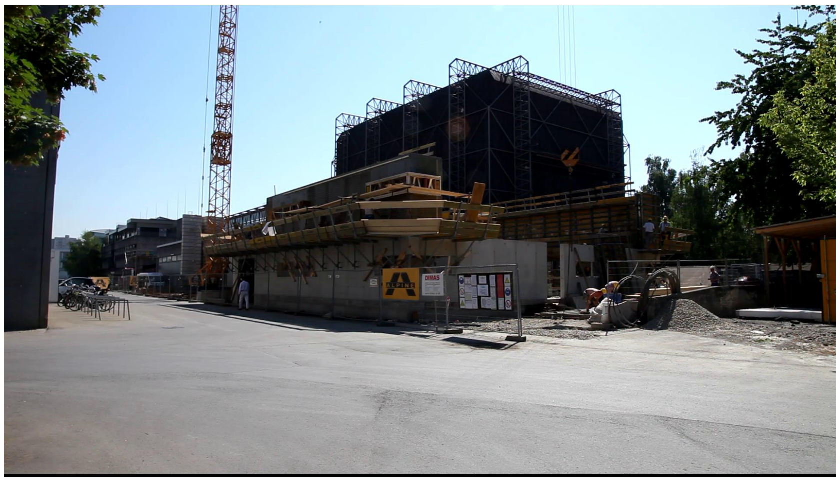
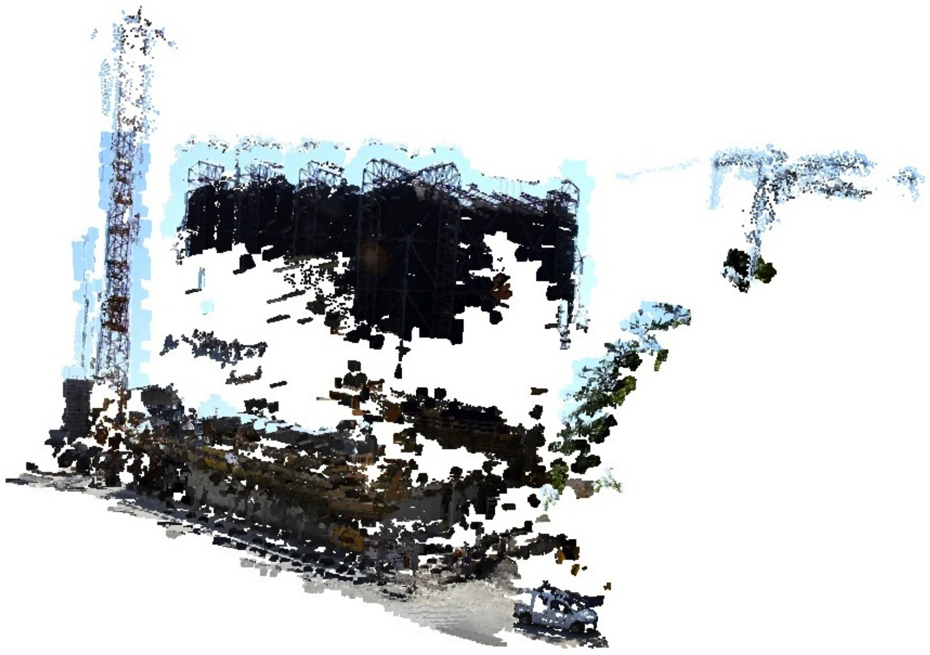


Figure 4.1: Brief overview of the 3D video rendering process. First, the video sequence is segmented into consistent regions with a state-of-the-art segmentation algorithm. Next, all 3D model points are projected onto the pre-localised video sequence resulting in an estimated depth value for each pre-segmented superpixel. Afterwards, each superpixel is projected back into the 3D space resulting in a realistic 3D video sequence.

At this point, we assume that the video sequence is already localised in the world coordinate system of the corresponding 3D model. Therefore, Equation (2.3) can be applied, which projects a 3D point X onto a 2D point x on the image plane. The projected points are undistorted with the previous estimated radial distortion parameters, as addressed in section 3.3.2. The Euclidian distance from the camera centre to the point X is assigned to the superpixel that the point x falls into. If several 3D points X are projected into one superpixel these depth values are accumulated and subsequently robustly averaged. More precisely, after the robust averaging of all distances projected into one segmented area, each superpixel is assigned a depth value, representing an average distance between the camera centre and the corresponding pixels in the 3D model.
Superpixels without corresponding points in the 3D model are not assigned depth value;



(a)



(b)

Figure 4.2: a) 2D video frame of a construction site. b) 3D video rendered from the 2D video sequence from a) by applying our 3D video rendering workflow.

hence, these regions cannot be used for the 3D video rendering process.

$$\mathbf{r} = K^{-1}\mathbf{x} = \begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix} \tag{4.1}$$

$$\begin{aligned} r_x &= -r_x \frac{d}{r_z} \\ r_y &= -r_y \frac{d}{r_z} \\ r_z &= -d \end{aligned} \tag{4.2}$$

$$\mathbf{X} = R^T(\mathbf{r} - t) \tag{4.3}$$

To summarise, at this point every superpixel with a corresponding 3D point is assigned an average depth value. Now, this depth information is used to project the segmented regions into the space leading to a realistic 3D video sequence. One image point $\mathbf{x} = (x_i, y_i, 1)$, written in homogenous coordinates, is projected into a 3D point $\mathbf{X} = (x, y, z)$ through applying Equation (4.1) to (4.3). First, Equation (4.1) computes a ray from the camera centre to the space in the direction of the point X. Equation (4.2) normalises the ray, where d denotes the Euclidian distance from the camera centre and finally Equation (4.3) determines the projected point in the 3D space referring to the world coordinate system. Figure 4.2b shows a snapshot of such a 3D video sequence. In addition, to achieve a more realistic appearance of the 3D video the colour of the corresponding pixels in the 2D video frame is rendered onto the 3D video as one can see in this figure. For comparison purposes, the original 2D video is shown in Figure 4.2a. Moreover, a sparse and a dense 3D model of the same construction site reconstructed with approximately 80 images and modern SfM approach has already been shown in Figure 2.16.
Since both, the 3D model and the back projected video, can be rendered into the same scene, changes between the original reconstructed 3D model and the computed 3D video sequence can be detected visually. Furthermore, regions without a correspondence between the reconstructed model and the videos (e.g. sky) are detected automatically without further knowledge of the geometry of the scene.

4.2 Video Segmentation

As addressed in the previous section, the 3D video rendering process requires some sort of segmentation; hence, this section introduces some basic ideas about image and video segmentation. First, some general information regarding image segmentation is given.

Next, several image and video segmentation algorithms are addressed and finally the outcome of such a state-of-the-art algorithm is shown.
As addressed by Zhang [122] image segmentation has a long history and can be traced back more than four decades. Since then, a great deal of research has been done resulting in several discriminative segmentation algorithms which can be classified into five major categories

- Pixel-based Segmentation
- Edge-based Segmentation
- Region-based Segmentation
- Model-based Segmentation
- Texture-based Segmentation

but the borders between the several categories are not clearly defined. Several methods from different classes are sometimes combined to advance the outcome of the algorithms. Furthermore, we should mention that several other classifications are presented in literature, but most of the segmentation algorithms fall into the five classes addressed here.

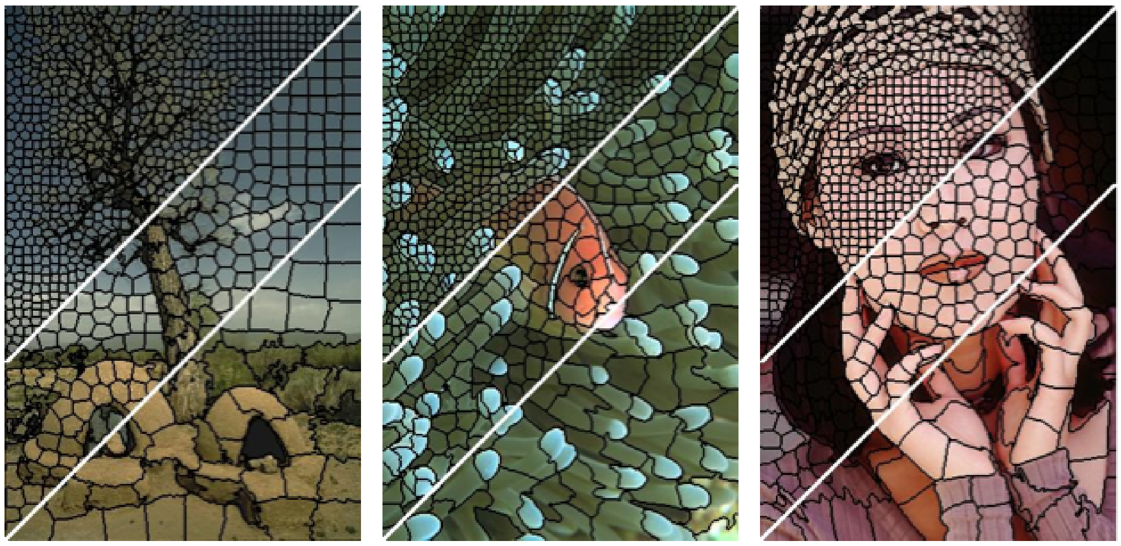


Figure 4.3: Comparison of various segmented images after applying the SLIC superpixel approach for different parameters[1]. The number of superpixels can be easily specified by a single parameter.

Some of the well-known segmentation algorithms are *Thresholding, Histogram-based methods, Region growing, split-and-merge, watersheding methods* and *clustering methods* to

[1]These images are taken from http://ivrg.epfl.ch/supplementary_material/RK_SLICSuperpixels/index.html

mention only a few. Since the fully-automatic image segmentation algorithms were sometimes lacking in terms of accuracy, several semi-automatic segmentation algorithms have since been published. Nevertheless, in this master's thesis a fully-automatic segmentation algorithm is chosen. Morever, Malisiewicz and Efros [63] have shown in their work that no optimal segmentation algorithms exist. However, they have empirically proved that multiple segmentation approaches perform better than a single segmentation. Such a multiple segmentation approach is the work proposed by Kluckner et al. [53], which additionally incorporates contextual constraints. This approach would significantly enhance the segmentation; however, since the performance is an important issue, as videos rather than images are processed, this approach is not applicable.

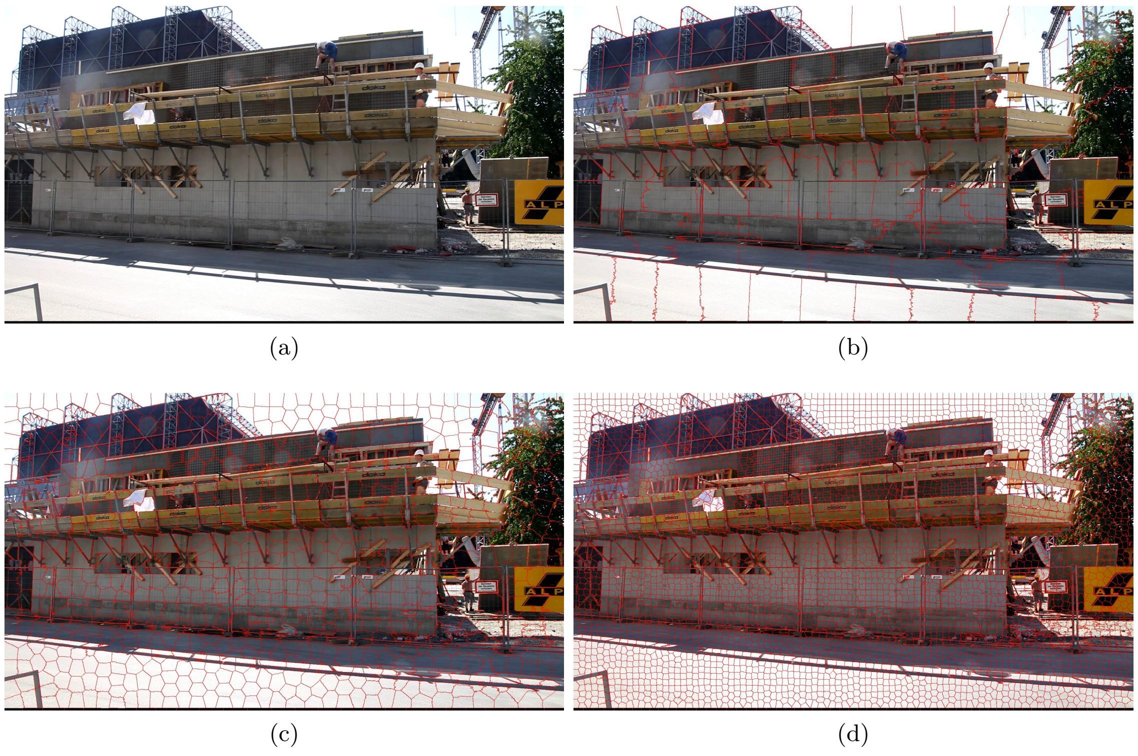

(a) (b) (c) (d)

Figure 4.4: Original video frame versus videos segmented with the SLIC superpixel approach. The number of superpixels can be easily specified by a single parameter, which is set to b) 100, c) 1000 and d) 5000.

Recently, Lucchi et al. [62] proposed a novel image segmentation technique to determine consistent regions through an iterative pixel clustering approach. This method is termed Simple Linear Iterative Clustering (SLIC) and had a major breakthrough two years ago, since it performs well on various images. The Quick shift image segmentation algorithm, developed by Vedaldi and Soatto [111], is another widely-known and fast segmentation algorithm. Furthermore, Comaniciu and Meer [19] proposed a novel non-parametic segmentation method by applying the well-known mean shift procedure. Other recently

developed image segmentation algorithms were proposed by Felzenszwalb et al. [24], Mori [69] and Levinshtein et al. [55] to mention a few.
Due to the modularity of our approach, each of these image segmentation algorithms could be used; however, we decided to segment our images with the SLIC superpixel approach, as executables are freely available and the number of superpixels can be easily defined by only one parameter. Moreover, as addressed in the paper, this approach outperforms several state-of-the-art image segmentation algorithms regarding boundary recall and under-segmentation error, while the computation time is reduced.
Figure 4.3 shows some fine and rough results after segmenting images with the SLIC superpixel approach for various parameters. Additionally, Figure 4.4 illustrates an original photograph of a construction site (see Figure 4.4a) and segmented images of the same photograph for different amounts of superpixles after applying the SLIC approach. The number of superpixels can be defined by a single parameter, which is set to 100 in Figure 4.4b, to 1000 in Figure 4.4c and to 5000 in Figure 4.4d.

4.3 Further Visualisations and Applications

In this section we will present two more applications, which use the previous computed depth values in another visualisation. The first application simply computes a so termed depth map. Depth maps are grey-scale images, where the pixel intensity corresponds to the Euclidian distance between the camera centre and the corresponding 3D point. The second application uses a novel approach to detect changes between the reconstructed 3D model and the registered video sequence.

4.3.1 Depth Map

In this visualisation the depth information for each video frame is represented in so called depth maps. In general, depth maps are grey-value images, where each grey value represents a different depth value. In our representation dark regions are closer to the camera centre, while brighter regions are further away, but this can be done the other way round as well. Moreover, we should mention that superpixels without corresponding 3D points are represented as black pixels in this visualisation. Figure 4.5 shows such a depth map for a video frame taken from our dataset. The original video frame (compare Figure 4.2a) from which this depth map was computed is the same as the one used for rendering the 3D video in Section 4.1.
The validity of this visualisation is proven as one can simply compare the distance from the viewer to the crane on the left part of the original video frame to the huge cube-shaped building in the middle. It is obvious that one cannot easily determine which object is closer to the viewer. On the other hand, if we take a look at the depth map image it is clear that the crane is closer, since this region is darker in comparison to the cube-shaped building in the middle.

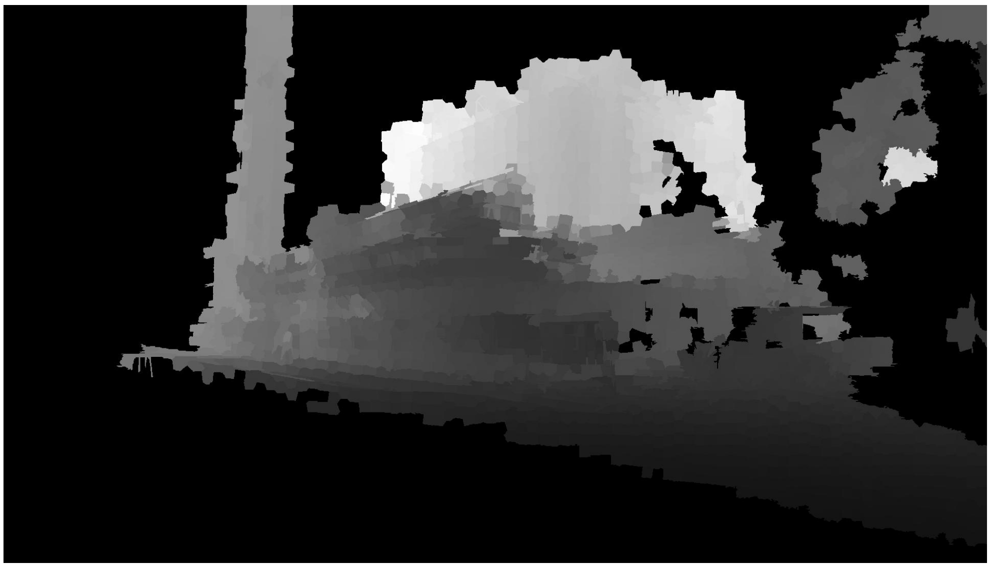


Figure 4.5: Through assigning various grey values to different depth values a depth map can be created. Dark regions are closer to the camera centre, while brighter regions are further away. Regions with no correspondence between the 3D model and the video frames (e.g. sky) are set to black.

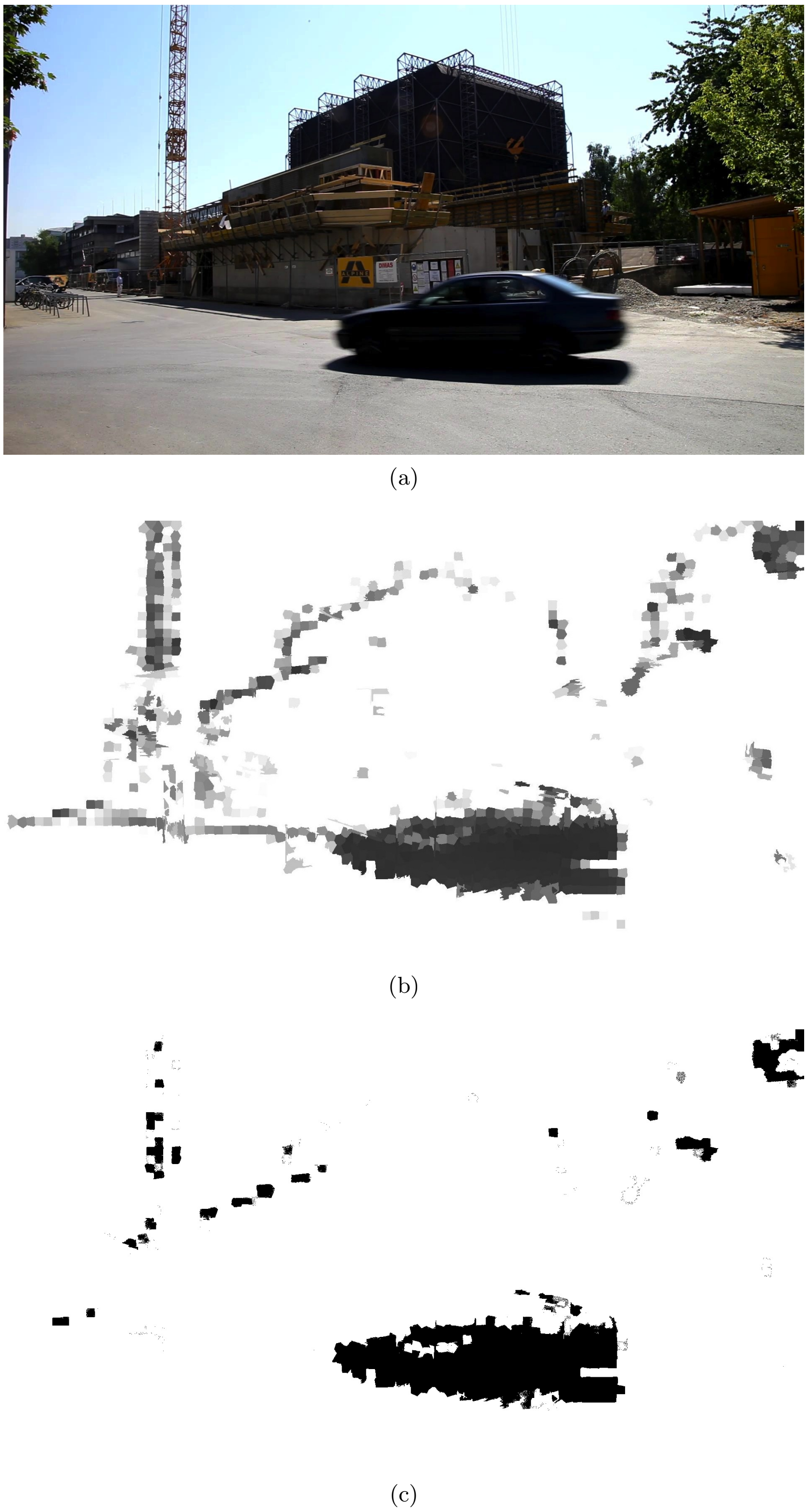
In our approach each video frame has to be registered independently which leads to a slight overhead in the 3D video rendering process. A more efficient way to propagate the depth information is to use such depth map images as an input to other processing pipelines such as recently evolved depth propagation approaches, which propagate the depth information from one frame to another leading to a 3D video as well.

4.3.2 Change Detection

Another way to use the estimated depth values is to detect changes between the reconstructed 3D model and the corresponding video. Through a standard back projection the position of the 3D point in the video can be determined. Since the colour of the 3D point is known a simple way to detect changes is to define a similarity function for the colour difference, as it is done in Equation (4.4). For each superpixel a ratio r_j is computed over all 3D points X_i which are projected onto superpixel j. N denotes the number of points projected into superpixel j. Moreover, $R(x)$ returns the red colour component, $G(x)$ the green one and $B(x)$ the blue one.

$$r_j = \frac{1}{N} \sum_{\forall X_i \in S_j} \sqrt{(R(X_i) - R(x_i))^2 + (G(X_i) - G(x_i))^2 + (B(X_i) - B(x_i))^2} \tag{4.4}$$

If the colour of the 3D point and the corresponding pixel in the video frame is similar, then this similarity function returns a very small value; otherwise a greater value is returned.



(a)

(b)

(c)

Figure 4.6: a) Single video frame of a construction site. b) Depth map for the video from a) after applying our change detection approach. c) Binary image after applying a simple thresholding technique with a threshold of 0.5 on the grey-scale image from b)

For simplicity each superpixel ratio value r_j is normalised to the range $[0, 1]$ through applying Equation (4.5).

$$CD_j = \frac{1}{1 + r_j} \tag{4.5}$$

Figure 4.6b shows the outcome for one video frame, whereas a high difference between the video and the 3D model is illustrated as a darker colour and a similarity with a brighter colour. Another possibility for visualising changes is to introduce a simple thresholding technique resulting in a binary image. Figure 4.6c illustrates such a binary image after thresholding Figure 4.6b with a threshold of $t = 0.5$.
Moreover, we should mention that a standard computer vision algorithm could be directly applied to the 2D video sequence to detect changes. This would work perfectly as well; nonetheless, using our approach we do get some additional depth information for the changed object. We can, therefore, perform a type of occlusion handling, which would not be possible with standard computer vision algorithms. Finally, we should mention that this approach only works if a designated colour difference between the 3D model and the registered video frame is present. Furthermore, this approach is not very robust against illumination and brightness variations, since it simply computes the colour difference. However, a colour transformation into another colour space (e.g. HSV) eliminates these drawbacks.

4.4 Workflow

This section briefly summarises the entire workflow of this master's thesis which consists of various exchangeable modules as shown in Figure 4.7.

The workflow incorporates:

- Sparse 3D reconstruction (see Section 2.3.1)
- Densification of the sparse model (see Section 2.3.2)
- Feature selection of the 3D model
- Camera calibration and video localisation (see Chapter 3)
- Splitting video into video frames
- Feature extraction from the video frames
- Segmentation of the video frames into consistent regions (see Section 4.2)
- Estimation of a depth value for each superpixel through projecting the 3D point onto the image plane.
- Rendering of 3D video sequences (see Section 4.1)

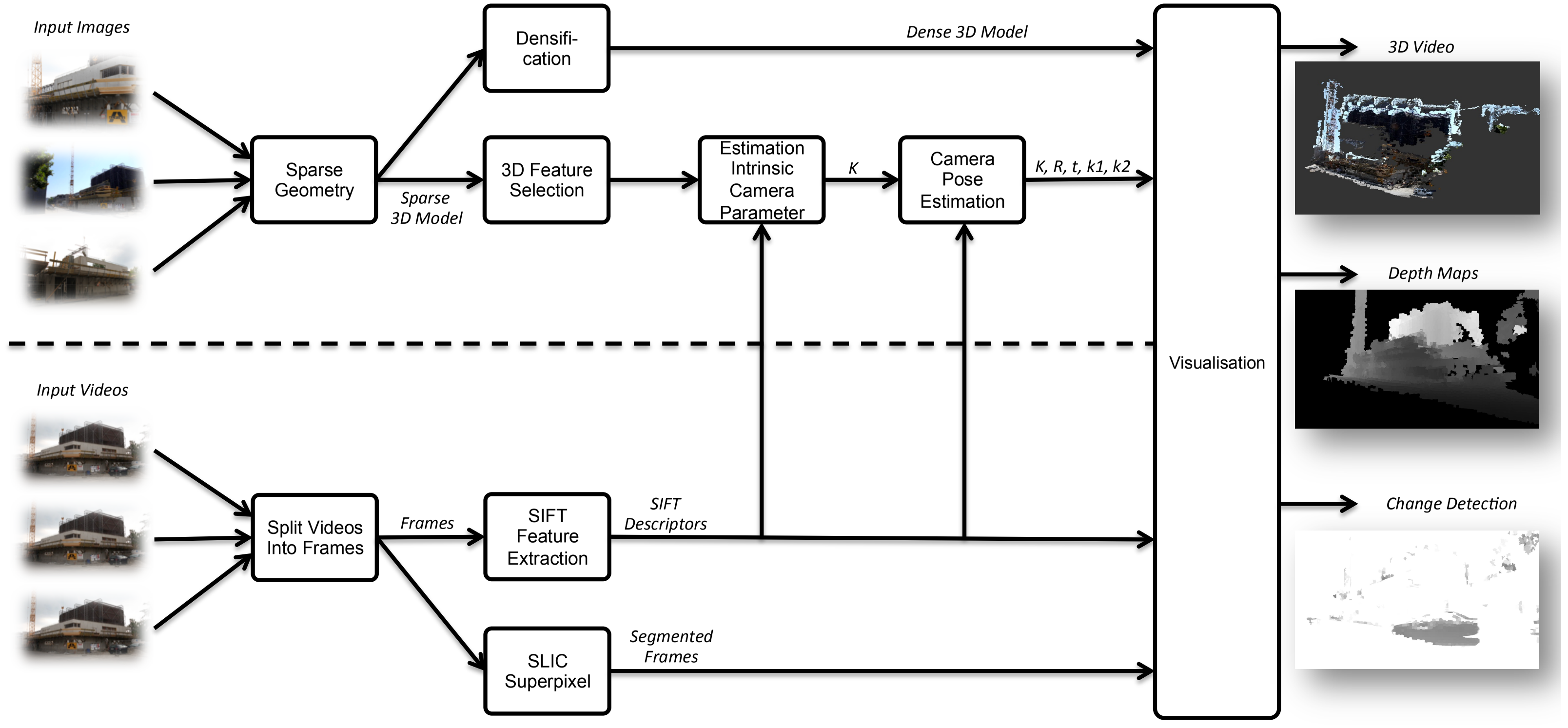


Figure 4.7: Flowchart of the entire workflow. First, a 3D model is reconstructed from the input images. Next, video frames are registered to this model. In the meantime the video frames are segmented into consistent regions. Afterwards, a depth value (Euclidian distance between 3D space point and camera centre) is assigned for each superpixel resulting in various visualisations as visible in the right side of the figure. 3D videos, depth maps and change detection are possible outcomes of this workflow.

First of all, some input data is required to render a 3D video. For reconstruction purposes, a couple of overlapping images from different viewpoints from the same scene are required, which can be easily accessed through Internet photo collection platforms. Additionally, in order to render a 3D video, or to detect changes, a video from the same scene is required. Once the process of downloading or capturing of input data is finished, the first step is to reconstruct a sparse model of the scene by applying an SfM approach as described in detail in Section 2.3.1. This sparse reconstruction is later used to create a denser model using the application of the techniques described in Section 2.3.2. In this master's thesis the dense models are only used to improve the appearance of the triangulated model, while the sparse model is later used as the basis for video localisation A reconstruction of such sparse and dense models is presented in Figure 2.16.
The output of the sparse model is a list including the 3D position $X = (x, y, z)$ based on the world coordinate system, the colour (RGB) and a feature descriptor for each triangulated point. The feature descriptor for each 3D point X is computed through a robust averaging and is denoted as *3D Feature Selection* in the flowchart. To improve the robustness of our approach and to minimise incorrect matches, only features which are visible in more than two images are used for further computations. With this step the pre-processing part, including sparse and dense 3D reconstruction and feature selection, is completed.
To register a video to a pre-computed 3D model, first the video is divided into single frames and subsequently the feature descriptors are computed for each frame. Through matching similar feature descriptors from the sparse 3D model and the video frames each frame is localised as described in Section 3.4.1. We should mention that each video frame is registered independently; therefore robust techniques to estimate the intrinsic camera parameters can be applied. The registration process provides the camera pose and additional robust intrinsic camera parameters, which can be used later to simplify the localisation process through applying a 3-point algorithm as addressed in Section 3.4.2 instead of the 6-point DLT.
Next, a standard segmentation algorithm (see Section 4.2) is applied to split video frames, which leads to consistent regions. These regions, which consist of several adjacent pixels, are also known as superpixels. In the next step the depth for each superpixel is estimated through a projection of the 3D points. More precisely, each 3D space point X is projected into the video frame and the Euclidian distance between the point X and the camera centre is assigned to that superpixel where the projected point X intersects with the image plane. This extracted depth information leads to additional information about the scene which can be helpful for several applications, such as 3D video rendering. In conclusion, this 3D video rendering process, together with computation of depth maps and a change detection approach has been comprehensively addressed in this chapter.

Chapter 5

Experiments and Results

Contents

This chapter evaluates our approach towards accuracy and robustness and presents results from some of our experiments. First, an overview of the software and hardware that was used is given. Second, the captured datasets, including images for 3D reconstruction purposes and 2D videos for 3D video rendering, are addressed. Additionally, specifications of the camera used are presented. Next, the evaluated results are shown. This includes the evaluation of the reprojection error, the estimation of the focal length on images and video sequences, the estimation of the camera pose and the influence of the radial distortion. Furthermore, the dependency between different datasets is illustrated through localising videos onto 3D models captured at different times. Finally, a comprehensive discussion of the results obtained is given.

5.1 Testing Environment

In this master's thesis the proposed platform for 3D video rendering is implemented in C/C++. The framework includes 3D modelling, feature extraction, feature matching, camera pose estimation, camera calibration, video segmentation and 3D video rendering and is platform independent. Furthermore, it has been compiled and successfully executed on Windows XP / Windows 7, Ubuntu 11.04 and on Mac OSX Lion 10.7 machines.
All evaluations in this chapter were done on an Intel Core 2 Quad @ 2.66 GHz with 4 GB of main memory and Ubuntu 11.04 Natty as an underlying operating system. The first version of the program is implemented to run on the central processing unit (CPU) and is not run-time optimised. Nonetheless, owing to the modularity of our approach, several parts of the workflow such as feature detection and 2D to 3D matching could be implemented on the graphics processing unit (GPU) to enhance the performance of the program. Therefore, in this section no run-time evaluations are done, except the comparison of the 3-point algorithm and the 6-point DLT, since these algorithms are both implemented on the CPU.

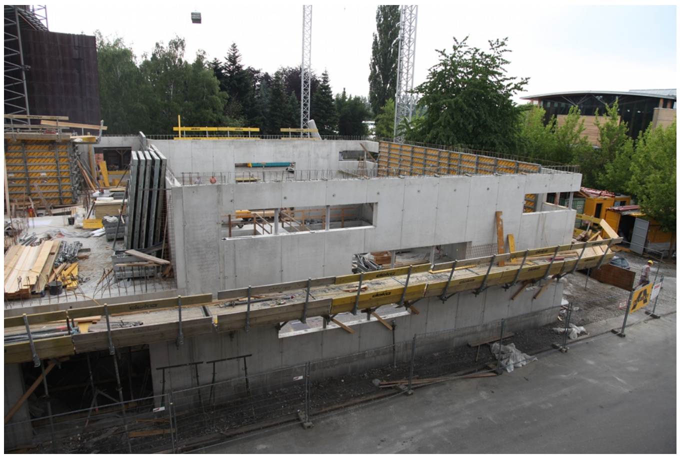
5.2 Input Data

The raw data used for evaluation purposes consists of twelve datasets distributed over nine days and various times of day. These datasets were captured at a construction

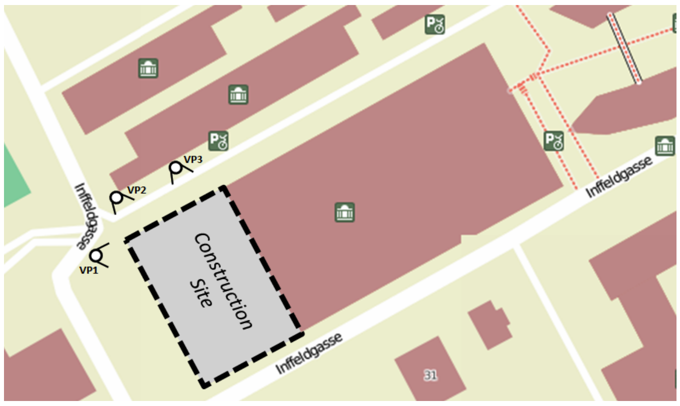
Dataset	Date	Time	#Images
1	2011-05-24	10:00	85
2	2011-05-24	12:00	83
3	2011-05-24	14:00	80
4	2011-05-24	17:00	79
5	2011-05-25	08:30	81
6	2011-05-25	10:30	81
7	2011-05-25	12:00	88
8	2011-05-25	16:30	77
9	2011-05-27	12:00	89
10	2011-05-30	18:00	77
11	2011-05-31	17:30	84
12	2011-06-01	11:00	77

Table 5.1: Overview of the date and time of the twelve captured datasets. On the first two days, four datasets were captured each day and thereafter only one dataset per day. The horizontal line separates each day.

site, which is located next to the ICG (Institute for Computer Graphics and Vision - University of Technology Graz) department. All images were captured without a tripod.



(a)



(b)

Figure 5.1: a) Photograph of the construction site captured from the 2nd floor of the ICG building, which is located on the opposite side of the construction site. b) The street map is taken from OpenStreetMap[1] and shows details of the TU Graz campus Inffeldgasse. The area of the construction site is shaded in grey and the videos were captured from the three viewpoints indicated on the map (VP1, VP2, VP3).

All evaluations in the following sections use these raw datasets, since no other appropriate datasets with ground truth data that fitted our intentions were available to us.
Table 5.1 shows the day of capture of the images and videos, respectively. On the first two days, four datasets of nearly uniform distribution over the day where captured in order to get datasets with a slight disparity. This is especially interesting as a simulation of whether images from one dataset are registrable to other datasets, since this can be important for change detection. Furthermore, the next four datasets were captured on different days resulting in a higher disparity between consecutive sets.

[1]http://www.openstreetmap.org

5.2.1 Images for 3D Reconstruction

Each dataset includes around 75-90 images and twelve additional videos, from three different viewpoints, as shown in Figure 5.1b. Both the images and the videos were captured

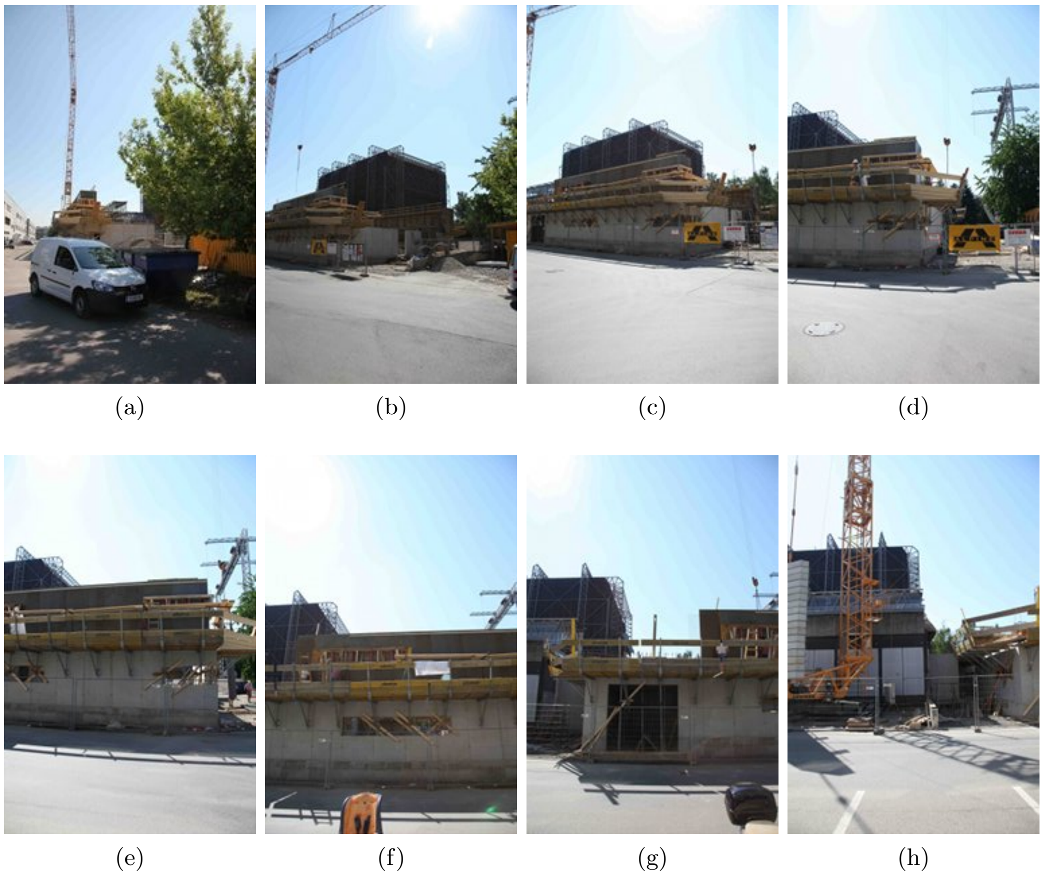


Figure 5.2: Selection of several images from the first dataset temporally ordered. The images were taken from two sides (north and west) of the construction site.

by a Canon EOS 5D Mark II. Table 5.2 lists some of the main specifications. A comprehensive user manual is available online[2]. The images were taken in portrait format with a distance of approximately one metre between two consecutive images. These images, which have a resolution of 5616×3744 pixels (21 Megapixel CMOS sensor), are used for the image-based reconstruction. An image selection of various viewpoints from the first dataset is shown in Figure 5.2. The images were captured from the north and west sides (see 5.1b) of the construction site. A reconstructed sparse 3D model and a corresponding dense model based on this dataset, which includes a total of 85 images, is shown in Figure 2.16.

[2]http://www.usa.canon.com/

Canon EOS 5D Mark II	
Resolution Images	5616×3744 pixels
Resolution Videos	1920×1080 pixels (Full HD)
CMOS Sensor	36×24 mm with 21.1 Megapixels
Shutter	1/8000 sec. to 30 sec.
Exposure	ISO 100 - 3200

Table 5.2: Selection of the main specifications of the Canon EOS 5D Mark II which was used for capturing all datasets including images and videos.

5.2.2 Video Sequences for 3D Video Rendering

In addition to the images, several videos were captured, which have a length of roughly 30 seconds each and a resolution of 1920×1080 pixels (Full HD). From each viewpoint four different videos were taken, which differ only in the camera properties. In the first case the camera is static (A), in the second case we allow rotation (B) and in the third case the camera rotates and changes the focal length (C) during the capturing process. The last case has no constraints and allows translation, rotation and zooming (D). Since we captured twelve datasets and each contains twelve different videos we have in total 144 videos. Each video is approximately 30 seconds long; therefore the total material comprises about 72 minutes of video. Figure 5.3 shows the construction site viewed from the three different viewpoints.

5.3 Results

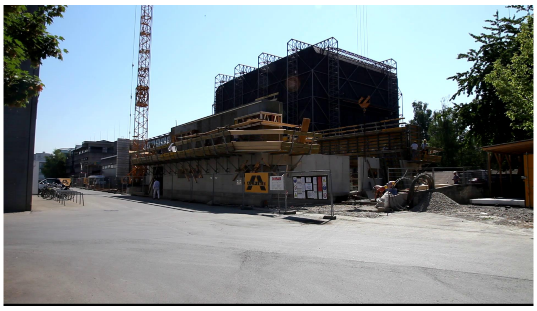
This section evaluates our approach in terms of accuracy and robustness by comparing it with other state-of-the-art approaches because no datasets with ground truth data were available to us. Since SfM and image-based localisation is a highly relevant research topic, several researchers have published their evaluated results, which will be used for comparison in some sections.

5.3.1 Reprojection Error

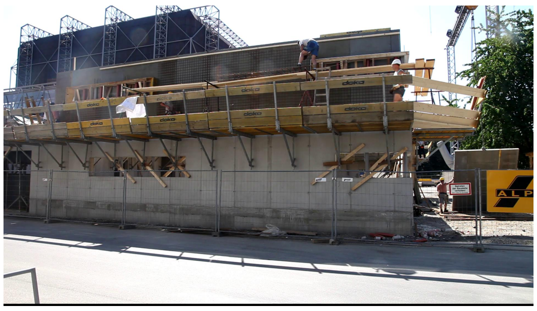
This section evaluates our approach in terms of reprojection error, which is defined in Equation (5.1), where n denotes the number of inliers and $f(x)$ is the undistortion function. The reprojection error E is given in pixels.

$$E = \frac{1}{n} \sum_{\forall inliers} |f(PX) - x_{true}| \tag{5.1}$$

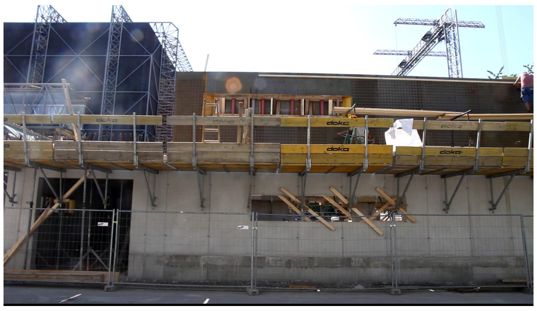
Table 5.3 illustrates the reprojection error for three different viewpoints. Furthermore, the mean reprojection error is computed for each dataset. For each viewpoint 100 video frames are localised and the average reprojection error is subsequently computed. This



(a)



(b)



(c)

Figure 5.3: Photographs from the three different viewpoints. a) Viewpoint 1, b) Viewpoint 2, c) Viewpoint 3.

is important for obtaining reasonable results, since the RANSAC framework randomly selects the first inlier set in each iteration.

From this evaluation we can conclude that the reprojection error is at most 1.52 *pixels*. Moreover, the reprojection error over all twelve datasets is, on average, less than 0.96 *pixel*. In conclusion, as presented during this evaluation, the reprojection error of our approach is in the sub pixel dimension, which is sufficient for our purpose.

	Reprojection Error [pixel]			
Dataset	Viewpoint 1	Viewpoint 2	Viewpoint 3	Mean
1	0.62	1.09	0.75	0.82
2	0.59	0.94	1.08	0.87
3	0.62	0.91	0.87	0.80
4	0.55	1.31	0.95	0.94
5	1.13	0.58	1.02	0.91
6	0.62	0.54	0.75	0.64
7	0.43	0.84	0.88	0.72
8	0.70	1.11	0.61	0.81
9	0.57	1.26	1.40	1.08
10	0.54	0.95	1.78	1.09
11	0.58	1.34	2.64	1.52
12	0.82	1.65	1.27	1.25

Table 5.3: Reprojection error for all twelve datasets and different viewpoints.

5.3.2 Evaluation of the Focal Length on Images

This section compares the focal length computed with our localisation approach with three other methods. First, Exif information is extracted from the image files themselves. These Exif tags contain useful information about the image, such as manufacturer, camera model, image resolution, focal length and sensor size to mention but a few. As the focal length and the sensor size are given in mm an equation to transform the focal length to pixel units is required. Equation (5.2) transforms the focal length (mm) given by the Exif tags to a focal length in pixel units.

$$focal\,length\,in\,pixels \;=\; image\,width\,in\,pixels\;\times\;\frac{focal\,length\,in\,mm}{CCD\,width\,in\,mm} \tag{5.2}$$

All three equation parameters, which are *image width*, *focal length* and *sensor size*, are available in the Exif tags; however, the charge-coupled device (CCD) width is not very accurate as addressed by Snavely. Therefore, this information should be taken from the user manual of the camera. For the Canon EOS 5D Mark II, which was used to capture all the datasets, the CCD width is given by $36.0mm$ as illustrated in Table 5.2. Finally, applying Equation (5.2) results in a focal length in pixel units of $f = 5616 \times \frac{24.0}{36.0} = 3744$ $pixels$.

In the second method we compare our results with the Photo tourism approach published by Snavely et al. [94]. This state-of-the-art approach estimates the intrinsic camera parameter and the pose of each camera during the 3D iterative reconstruction process. In each iteration one image is added and, simultaneously, the intrinsic camera parameters and the camera pose is estimated.

In the third method we compare our results with the approach proposed by Irschara

	Focal Length [pixel]		
	Median	Mean	Standard Deviation
Exif (see Equation (5.2))	3744	3744	0
Photo Tourism (1853×1236) [94]	3812.1	3822.1	81.5
Irschara et. al. (5616×3744) [45]	3807.52	3807.52	not available
1853×1236 (100%)	3807.9	3807.0	80.2
1390×927 (75%)	3806.1	3829.4	80.6
927×618 (50%)	3785.5	3818.2	83.6
463×309 (25%)	3795.8	3798.2	115.2

Table 5.4: Comparison of the focal length for different approaches including Exif, Photo tourism, Irschara's approach and our localisation approach for different image resolutions. For down-sampled images the focal length is multiplied by the down-sampling factor to get the focal length for the full image resolution.

et. al. [45]. This work aims to estimate a calibration with the accuracy of a target calibration technique not requiring precise calibration patterns. The method is based on distinguishable printed markers, which are deployed in an arbitrary fashion.
Basically, the focal lengths obtained by the Exif tags and the estimated focal lengths determined by the Photo tourism and Irschara's approach are compared to our approach. Since one requirement of our approach was to locate images with various resolutions, an evaluation for four different image resolutions is given.
Furthermore, we should mention that we used the SIFT implementation of David Lowe [60, 61], which restricts the image resolution to a size smaller than 2000×2000 pixels; however, other implementations which can compute the SIFT features at the full resolution, such as SiftGPU [115] can be used instead. Because of the restriction of Lowe's SIFT detectors the original images with a resolution of 5616×3744 are first down-sampled by a factor of three to an image resolution of 1853×1236. The images are down-sampled with a standard linear interpolation technique. For further discussion, these down-sampled images are always referred to as original images (100%).
Since our implementation estimates the focal length for the x and for the y direction and the other approaches only determine one focal length for both directions, we simply take the mean $f = \frac{f_x+f_y}{2}$ of these two focal lengths for comparison purposes. All evaluation results, except Irschara's approach since this is a pre-calibration technique, are based on the 85 images from the first dataset. As the image resolution directly influences the focal length in pixel units (see Equation (5.2)) all down-sampled images are multiplied by the down-sampling factor leading to the focal length which would be obtained if the full resolution had been used.

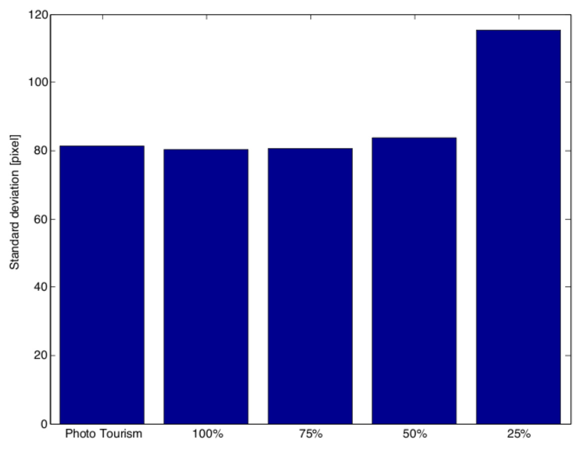


Figure 5.4: Bar graph representing the standard deviation for various image resolutions for our approach in comparison to the Photo Tourism approach.

The mean values are computed with

$$\bar{x} = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i. \tag{5.3}$$

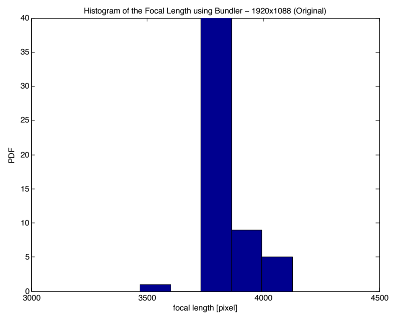
Moreover, the standard deviations are calculated applying

$$s = \sqrt{\frac{1}{n} \sum_{i=1}^{N} (x_i - \bar{x})^2}. \tag{5.4}$$

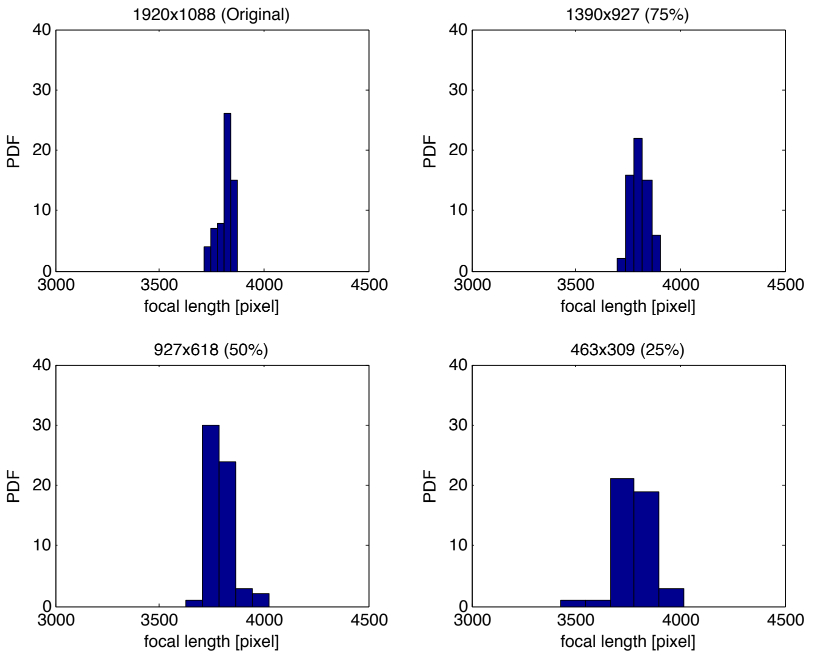
Furthermore, for the RANSAC implementation we assumed a maximum of 50% outliers and specified the probability of finding an inlier set with $p = 0.99$. A detailed explanation regarding the relevant RANSAC parameters and calculation of these values is given in Section 3.4.3.

Table 5.4 compares the focal length computed with our technique to the focal length obtained by the Exif tags of the images and the focal length computed with the Photo tourism and Irschara's approach, respectively. From this evaluation one can notice that our approach works well across various image resolutions. For high resolutions (1853×1236, 1390×927, 927×618) the mean deviation of the focal length between the Photo tourism approach and our approach is less than 0.5%. For lower image resolutions (463×309) the deviation increases, nevertheless it is $< 1\%$. Since images with a lower resolution lead to fewer SIFT keys the run time of the matching process and particularly of the pose estimation process are accelerated, but as evaluated the accuracy decreases.

Figure 5.4 shows the standard deviation values presented in Table 5.4 in a bar graph. While the standard deviation of the focal length for image resolutions down-sampled up to 50% is more or less equal to the standard deviation derived from the Photo Tourism approach, it increases significantly for images down-sampled by a factor of four.



(a)



(b)

Figure 5.5: Standard deviation of the focal length for
a) Photo tourism approach
b) our localisation approach for different image resolutions.

Another method of representing the robustness of our approach is to plot the focal length estimated for the images in a histogram. Figure 5.5 shows such a representation. We should mention that the localisation process fails for some images with a resolution of 463×309 (25%), hence some outliers are present. These outliers, which are the reason for the higher standard deviation in comparison to the three other image resolutions, are not represented in the Figure, since one scaling is used for all plots.

5.3.3 Evaluation of the Focal Length on Video Sequences

This section evaluates the estimation of the focal length for one of the video sequences, captured from viewpoint 1. The video sequence was taken from the first dataset. Table

	Focal Length [pixel]		
	Median	Mean	Standard Deviation
1920×1088 (100%)	3822.5	3817.4	36.2
1440×816 (75%)	3805.7	3804.2	42.8
960×544 (50%)	3784.4	3797.2	58.6
480×272 (25%)	3779.6	3781.2	95.1

Table 5.5: Comparison of the focal length of a video sequence for various resolutions. It is clear to see that the accuracy (standard deviation) increases as resolution decreases.

5.5 represents the estimated focal length for different video frame resolutions. It is clear to see that the standard deviation increases as the image resolution decreases, nevertheless the computed mean and median value still have a satisfying accuracy. The reason for the increasing standard deviation is the interpolation in the down-sampling process leading to different SIFT features. The difference between the evaluation in this section and

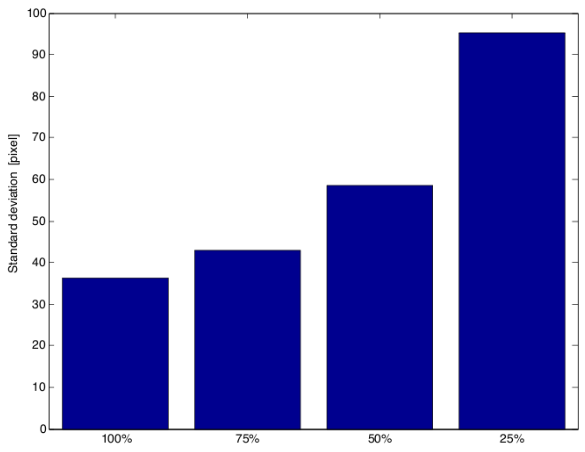


Figure 5.6: Bar graph representing the standard deviation of our approach for different video resolutions.

the evaluation presented in Section 5.3.2 is the different number of images/video frames. While in the previous case 85 images were registered, in this case only the first 45 frames of a video sequence were located. Another representation of the standard deviation of the focal length in the form of a bar chart is shown in Figure 5.6.
As the video sequence used in this section has a lower resolution (1920×1088) in contrast to the original images (5616×3744), the estimated focal length has to be divided by a

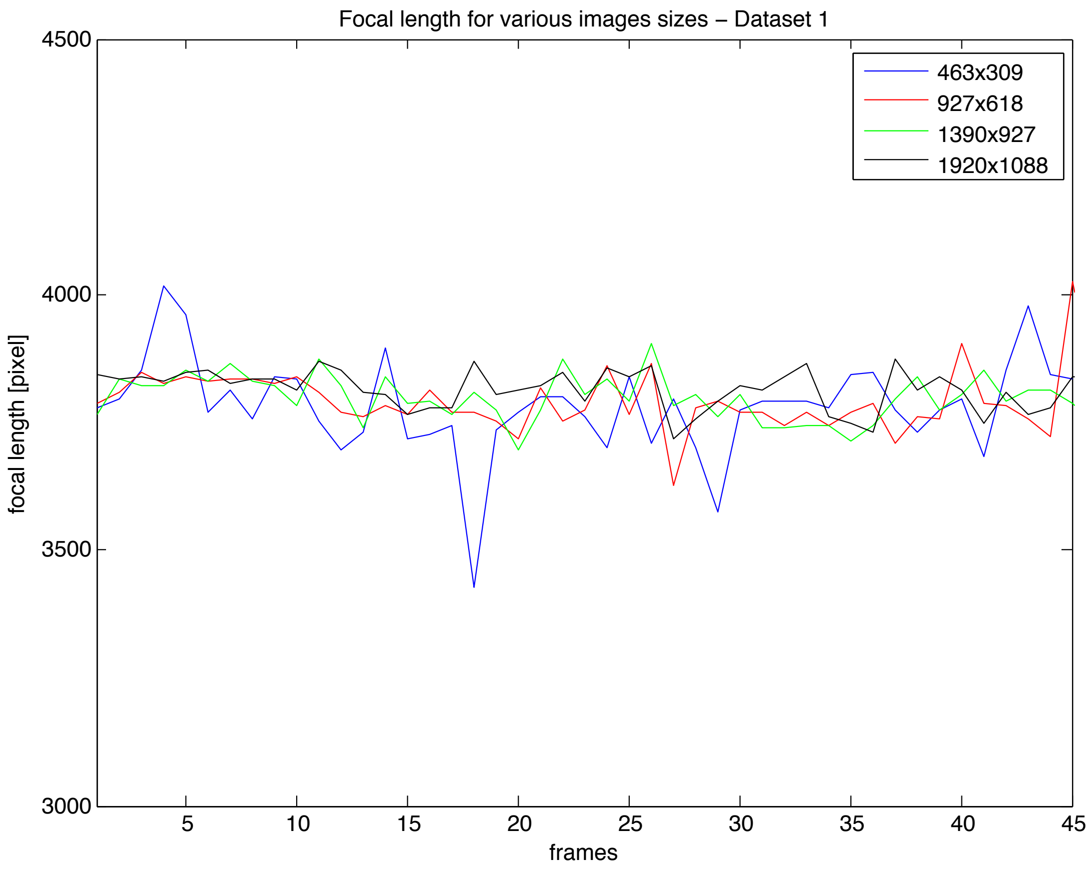


Figure 5.7: Estimation of the focal length for the first 45 frames of a video sequence from dataset 1 for different resolutions.

factor of $\frac{1920}{5616} = 0.341$. This is required to get the focal length that corresponds to the full image resolution. This additional multiplication step is required to obtain comparable values.

Furthermore, Figure 5.7 represents the variation of the estimated focal length for different video frames in one chart. As already addressed, one can see that videos with a high resolution result in smoother graphs, while decreasing video resolutions lead to higher variations in the estimated focal length.

5.3.4 Evaluation of the Camera Pose on Images

This section evaluates the accuracy of the camera pose compared to the position estimated through the reconstruction with the Photo tourism approach by Snavely et al. [94], which serves as comparable ground truth data for this evaluation. Table 5.6 shows the evaluated results. As a consequence of the Euclidian reconstruction the scale of the model is unknown; therefore, to get reasonable results the scale of the model is computed first. This is

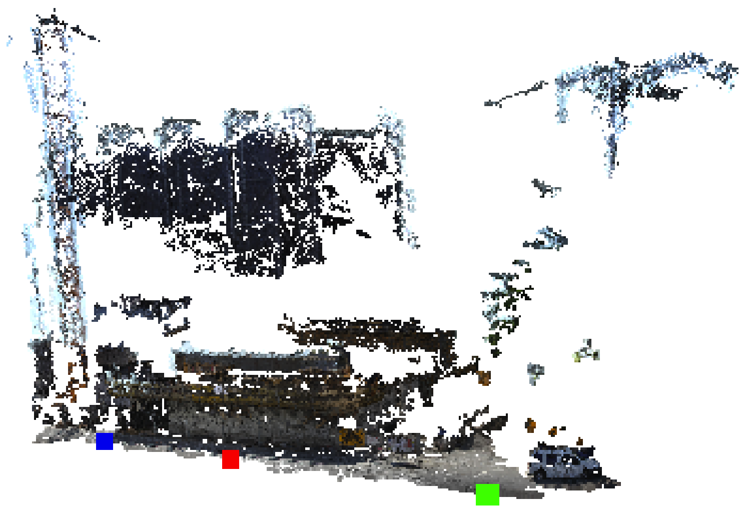
Dataset	#Images	#3D Points	Mean Deviation [m]
1	85	19082	1.61e-2
2	83	23538	8.70e-3
3	80	21961	3.36e-2
4	79	35874	4.47e-2
5	81	19701	7.54e-2
6	81	24728	8.40e-3
7	88	40225	5.28e-2
8	77	20286	4.40e-3
9	89	34802	7.93e-2
10	77	22226	7.93e-2
11	84	18829	9.88e-2
12	77	24162	7.18e-2

Table 5.6: Evaluation of the accuracy of the camera pose compared to the position estimated through the Photo tourism [94] reconstruction process. The column *#Images* denotes the number of localised images, while *#3D Points* are the number of reconstructed points. Finally, the *Mean Deviation* is calculated, which is averaged over all images from one dataset.

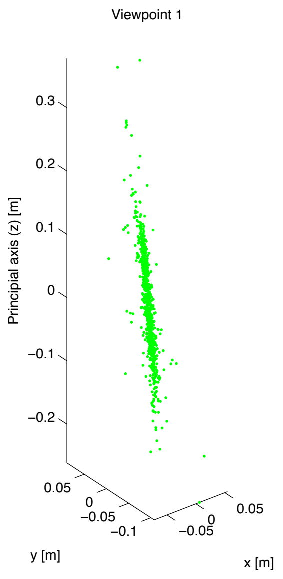
done by computing the maximum camera distance in each dataset. The maximum camera distance denotes the maximum distance between two arbitrary cameras in the coordinate system of the reconstructed model. In general, this maximum distance is the distance between the position where the first image was taken and the position where the last image was captured. Since the first and the last images for all twelve datasets were captured from the same viewpoints the direct distance, which is 48 metres, can be measured. With this known real distance the given Euclidian model can be scaled to a metric model.
Furthermore, the same images, which were used for the reconstruction, are localised with our approach. Subsequently, the Euclidian distance between the position determined with our approach and the Photo tourism [94] approach is computed. Finally, this distance multiplied with the pre-computed scaling factor leads to the absolute deviation given in metre.
From this evaluation we can conclude that the mean deviation for all twelve datasets is at most $0.0988m$. Moreover, the mean deviation over all twelve datasets is less than $0.05m$. One reason for these marginal variances is the fact that different radial distortion models are used. While the Photo tourism [94] approach only estimates the two distortion parameters (k_1, k_2), our approach estimates the centre of distortion (u, v) as well.

5.3.5 Evaluation of the Camera Pose on Video Sequences

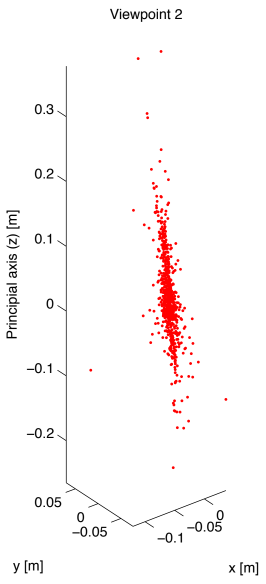
This section evaluates the deviation for various video sequences, which were captured from different viewpoints. For this evaluation the first dataset is used. Figure 5.8a shows



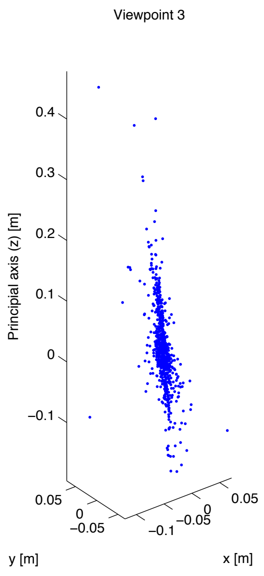
(a)



(b)



(c)



(d)

Figure 5.8: 3D plots of the localised video frames for the first dataset. a) Reconstructed model with the three different viewpoints, b) Viewpoint 1 (green), c) Viewpoint 2 (red), d) Viewpoint 3 (blue). As seen here, the estimated camera pose varies along the principal axes. Plots b), c) and d) are based on the particular camera centre.

the reconstructed 3D model together with the estimated camera pose for each viewpoint. Different colours mark the three different viewpoints - green for the first viewpoint, red for the second and blue for the third viewpoint. Based on the world coordinate system viewpoint 1 is located at $(X, Y, Z) = (0.31, 1.94, -11.26)$, viewpoint 2 at $(X, Y, Z) = (0.30, 5.99, -16.91)$ and viewpoint 3 at $(X, Y, Z) = (0.30, 8.64, -20.43)$.

Figure 5.8b to Figure 5.8d show the deviation for the three viewpoints in a detailed 3D scatter plot. We should mention that the evaluated model is scaled to a metric model, hence the scaling of the axes is given in metres. Furthermore, each of these three plots is based on the particular camera coordinate system. In this camera coordinate system each camera is located at $(X, Y, Z) = (0, 0, 0)$. While the standard deviation along the x and y axis is approximately $0.7cm$ and $1.3cm$, it is around $8.2cm$ for the principal axis. This fact is a consequence of the inaccuracies during the photogrammetric calibration.

	Standard deviation [m]		
Dataset	x	y	z
1	0.0072	0.0132	0.0820
2	0.0152	0.0290	0.1488
3	0.0225	0.0554	0.2013
4	0.0162	0.1037	0.1054
5	0.1475	0.0669	0.1837
6	0.0135	0.1598	0.0388
7	0.0758	0.0324	0.2010
8	0.0618	0.3636	0.2897
9	0.0409	0.0566	0.3105
10	0.0179	0.0535	0.1977
11	0.0494	0.2000	0.1492
12	0.0986	0.1539	0.1519

Table 5.7: Camera pose evaluation on full resolution video sequences captured from the first viewpoint. The computed standard deviation is given in metres.

In addition, Table 5.7 illustrates the standard deviation for all twelve datasets. As shown, the maximum standard deviation is less than $0.37m$ for all configurations. This is quite an accurate value, since no tripod was used during the capturing process. Therefore, minor inaccuracies were included during the capturing process of the video sequences.

5.3.6 Evaluation of the Radial Distortion

As addressed in Section 3.3.2 radial distortion is an issue with many standard digital cameras; therefore, introducing such a model can significantly enhance the accuracy of the resulting model. Figure 5.9 compares the estimated focal length for one video sequence from the first dataset without (a) and with (b) an additional radial distortion model. The lower resolution of the video sequence (1920×1088) leads to a focal length which is lower, by a factor of 0.341 (see Section 5.3.3), than the original focal length estimated for images with a full resolution of 5616×3744. As shown in Figure 5.9b introducing a radial distortion model leads to significantly smoother focal lengths for each frame compared to a model without an additional radial distortion model, which is shown in Figure 5.9a. The variations in the focal length around frame 300 are due to a fast pan in the video sequence, which resulted in a noticeable motion blur in the extracted video frames.

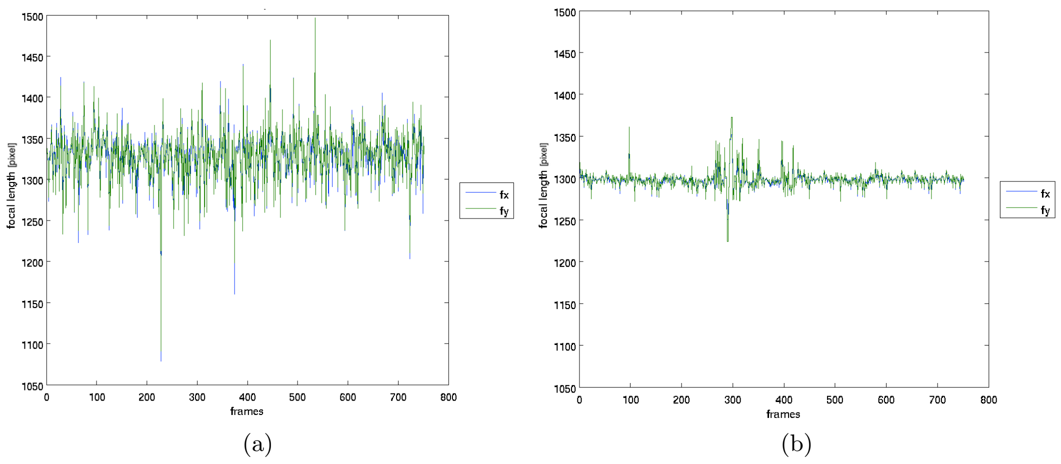
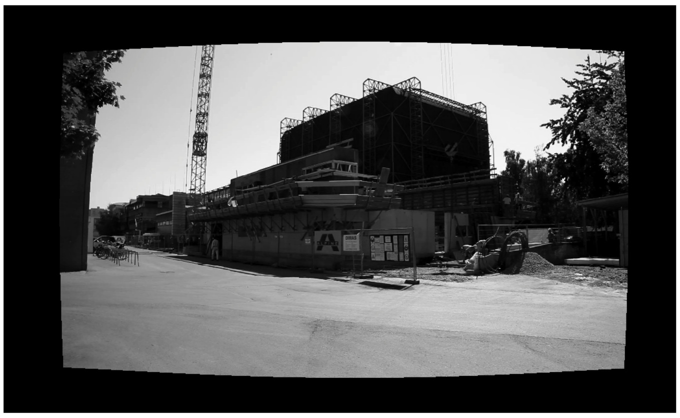
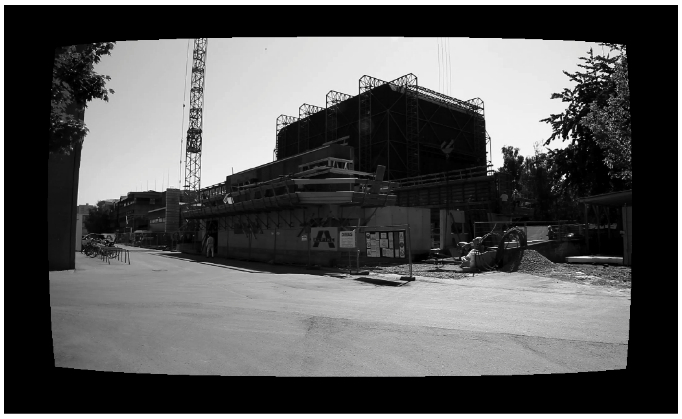


Figure 5.9: Focal length f estimated without (a) and with (b) an additional radial distortion model. The video sequence was taken from viewpoint 1 from dataset 1 and includes rotation but neither zooming nor translation.

The influence of the radial distortion on real images is shown in Figure 5.10. It compares a photograph undistorted with the parameters (k1, k2) estimated through the iterative 3D reconstruction applying the Photo Tourism approach (Figure 5.10a) to an image undistorted with our approach (Figure 5.10b). Both approaches are based on the radial distortion model addressed in Section 3.3.2, but are different in the number of estimated parameters. While the Photo tourism [94] approach assumes the centre of radial distortion is in the middle of the image and therefore estimates only the two distortion parameters $k1$ and $k2$, our approach also estimates the centre of radial distortion u and v. A visual comparison of the two undistorted images shows only a marginal disparity; still, further evaluations brought about small improvements through the additional parameters. Furthermore, another well known illustration of the radial distortion is what is known as the radial distortion map, shown in Figure 5.11. As this Figure shows, the radial distortion is relatively low for points near the image centre, whereas the radial distortion increases for points further away from the image centre.



(a)



(b)

Figure 5.10: Comparison of two undistorted images taken from the third dataset applying a) the Photo tourism [94] radial distortion model and b) our radial distortion model explained in Section 3.3.2

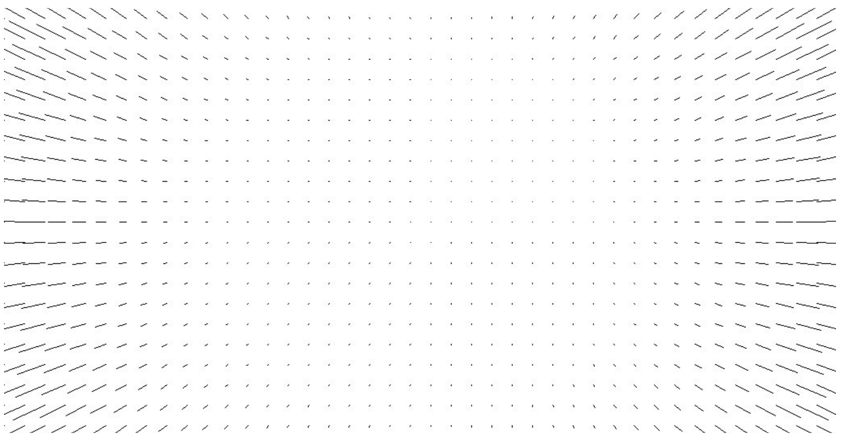


Figure 5.11: Radial distortion map for the image from Figure 5.10b applying our radial distortion model.

5.3.7 Run-time: 6-point DLT versus 3-point Algorithm

This section presents the run time improvements of the 3-point algorithm over the 6-point DLT. Due to there being fewer unknown parameters with the 3-point algorithm in comparison to the DLT algorithm a speed-up depending on the number of outliers is noticeable. Table 5.8 illustrates these enhancements. For a clearer interpretation, Figure

Fraction of outliers e	RANSAC k	RANSAC d	6-point [s]	3-point [s]
0.2	23	160	0.41	0.10
0.3	56	140	0.96	0.10
0.4	145	120	0.86	0.10
0.5	439	100	2.12	0.10
0.6	1683	80	4.66	0.10
0.7	9472	60	48.24	0.11

Table 5.8: Run-time comparison of the 6-point DLT and the 3-point algorithm depending on the number of outliers. The run-times are measured in seconds and include only the camera pose estimation step; the feature extraction and matching procedure are not included, since both approaches require the same set of 2D to 3D matches. Furthermore, the 6-point algorithm includes the additional radial distortion optimisation step. The two required RANSAC parameters k and d are determined through Equations (3.21) and (3.22) with $N = 200$ and $p = 0.999$.

5.12 shows the determined run time values in a graph. This chart illustrates that the

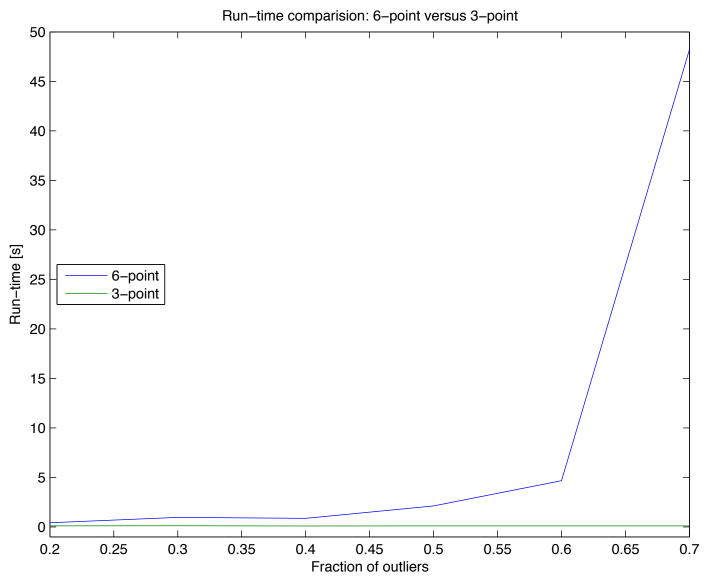


Figure 5.12: Run-time comparison of 6-point DLT to the 3-point algorithm.

run-time of the 6-point DLT increases with the number of outliers, while the run-time for the 3-point algorithm is more or less constant. For the run time evaluation in this section the first dataset was used, which consists of 18912 3D points. Furthermore, to advance the validity of this experiment the evaluation was repeated 100 times and robustly averaged, since the RANSAC loop selects the input points randomly. The empiric threshold value t (see Section 3.4.3) is set to $t = 2$, the number of iteration $N = 200$ and $p = 0.999$. The two required RANSAC parameters k and d are determined through Equations (3.21) and (3.22).
From these outcomes it is clear that calibration of the camera with a 6-point DLT and then registering the following video frames with a calibrated camera greatly speeds-up the localisation process by simultaneously preserving the accuracy of the camera pose. One reason for the slower run-time of the 6-point DLT algorithm is the fact that for each inlier set an additional refinement step is added. These refinement steps, together with the radial distortion estimation, have a great influence on the run-time of this algorithm.

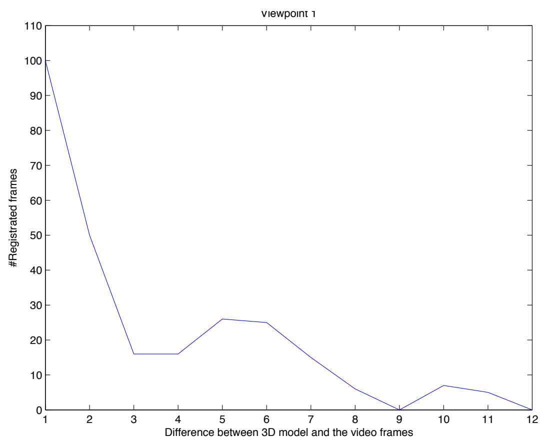
5.3.8 Registering Video Sequences onto Different 3D Models

In this section we try to register video sequences onto 3D models taken from different datasets. The time difference between the captured datasets vary from a few hours to

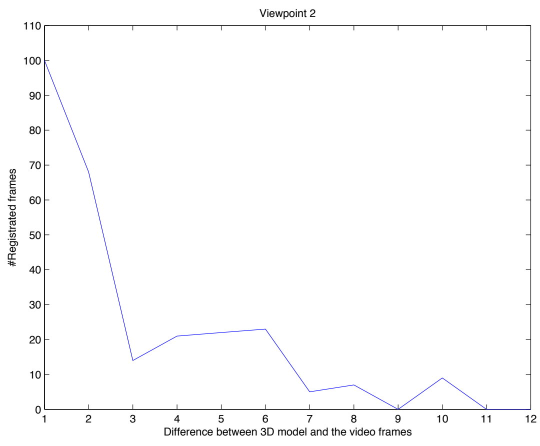
	#Registered Frames			
Difference	Viewpoint 1	Viewpoint 2	Viewpoint 3	Mean
0	100	100	100	100
1	50	68	62	60
2	16	14	43	24.3
3	16	21	50	29
4	26	22	30	26
5	25	23	33	27
6	15	5	17	12.3
7	6	7	15	9.3
8	0	0	0	0
9	7	9	0	5.3
10	5	0	0	1.6
11	0	0	0	0

Table 5.9: Number of registered frames between a 3D model and a video sequence, which was captured at a different time. For each dataset 100 images were used for localisation. The first column defines the difference between the datasets, while the last one represents an average number of localised video frames.

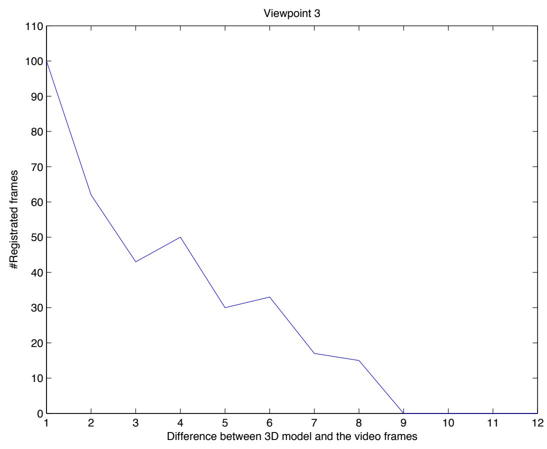
several days as addressed in Table 5.1. Table 5.9 presents the number of localised video frames. In each evaluation 100 video frames were used.
A graphic representation of the values represented in the Table 5.9 is shown in Figure



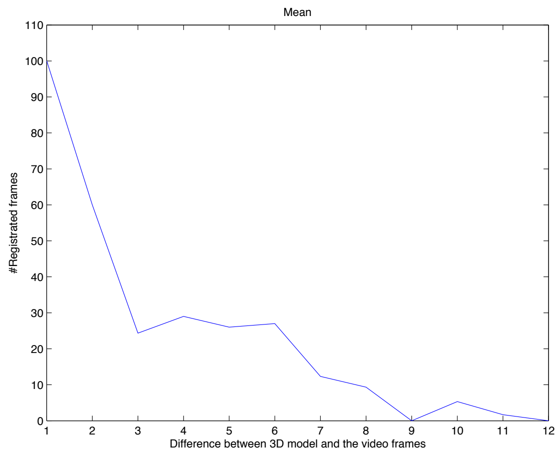
(a)



(b)



(c)



(d)

Figure 5.13: Number of registered frames between a 3D model and a video sequence captured at a different time for a) Viewpoint 1, b) Viewpoint 2, c) Viewpoint 3 and d) Mean of all three viewpoints. In total 100 images were used for localisation.

5.13. From these figures it is clear that the number of localised images decreases as the distance between the datasets increases. This is a consequence of the building process at the construction site, since the shape of the building changes over the course of time. These changes lead to diverse feature descriptors, which results in fewer localised frames. If the videos and the 3D model are taken from the same datasets all images are correctly localised.

In addition to this evaluation the reprojection error, as defined in Equation (5.1), is computed for all localised video frames and is shown in Figure 5.14 in what is known as a confusion matrix. From these figures we can see that 3D models and videos taken from the same dataset have the smallest reprojection error, illustrated as black squares in the

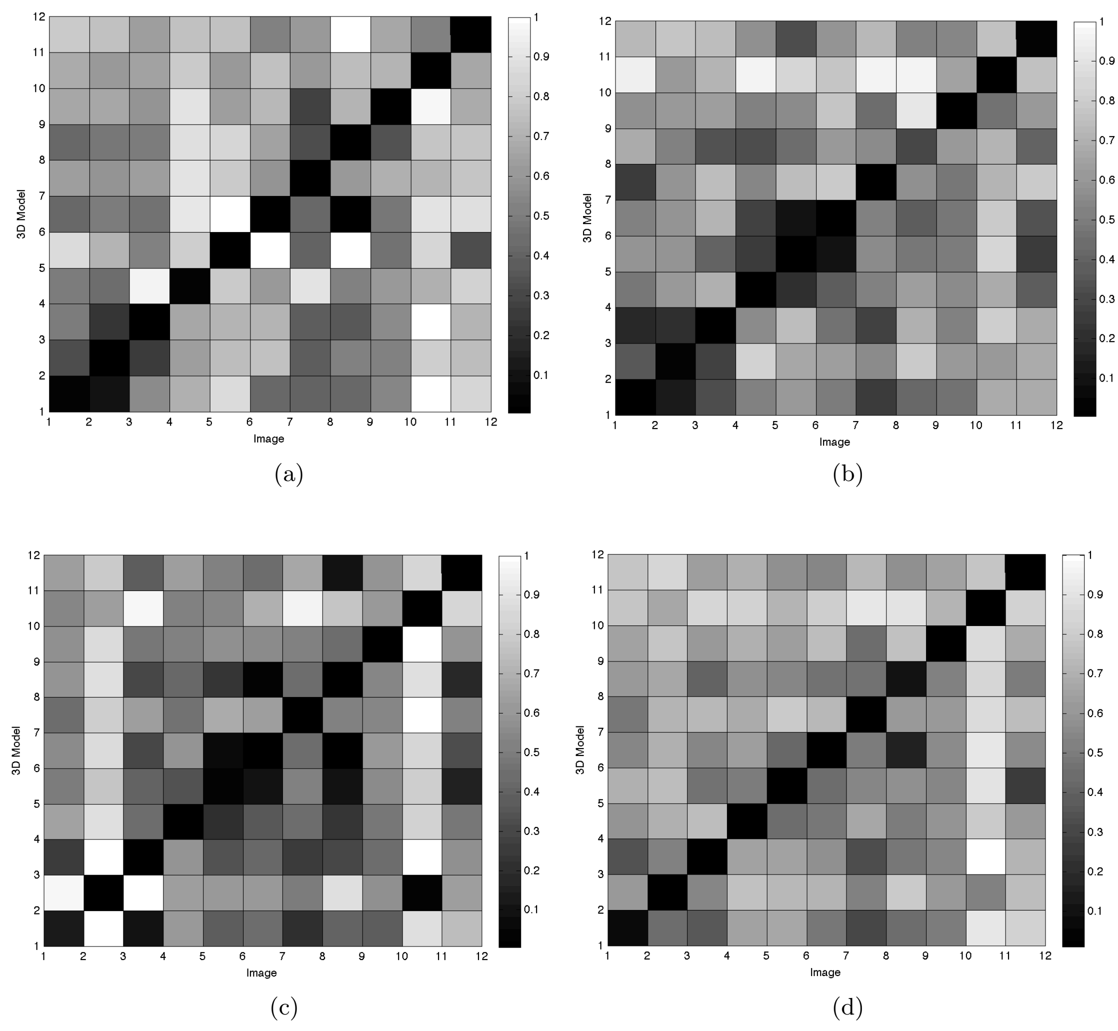


Figure 5.14: Normalised reprojection error between a 3D model and a video sequence captured at a different time for a) Viewpoint 1, b) Viewpoint 2, c) Viewpoint 3 and d) Mean over all three viewpoints. If the 3D model and the video sequence are taken from the same dataset the reprojection error is small, while an increasing distance between model and video leads to a bigger reprojection error.

chart. In conclusion, Figure 5.14d shows the mean reprojection error over all three viewpoints. As one can see, the top left and the bottom right corners are brighter representing a bigger reprojection error. These two corner squares have a distance of twelve, which means the model is taken from the first dataset and the video is taken from the last one and vice versa. Therefore, the reprojection error is much bigger in comparison to videos and 3D models taken from the same dataset, which are represented in the diagonal of these figures.

As listed in Table 5.1 the first four datasets were captured on the same day. This is clearly visible in Figure 5.14b. The 3×3 square in the bottom left corner is darker, which represents a smaller reprojection error in comparison to the brighter squares. The same is true for the second day, which also includes four datasets.

5.4 Summary

This section presented the evaluation results of our proposed approach in comparison to several state-of-the-art implementations.

The results obtained show that our approach is comparable with other modern approaches in terms of accuracy and robustness. Furthermore, several experiments on real datasets including images and video sequences led to acceptable outcomes, regarding accuracy of intrinsic camera parameters and camera pose. Moreover, experiments excluding a radial distortion model and including an additional introduced radial distortion model demonstrate a significant enhancement in regards to the robustness of the intrinsic camera parameters. However, the radial distortion is estimated with a non-linear optimiser which gives rise to additional run-time costs, since in every RANSAC step the radial distortion is estimated. A visual comparison of an undistorted photograph with the model applied by the Photo tourism approach and the model proposed in this thesis has brought about no considerable difference.

Finally, the dependency between the datasets was represented in the form of a confusion matrix, which evaluates the possibility of registering a video sequence on 3D models captured at different days and times of day. This experiment has shown that increasing time lag between the 3D model and the video sequence resulted in an increased reprojection error.

In conclusion, all results show that the proposed approach is comparable with other modern localisation approaches; nevertheless, including an additional radial distortion model, which is already in the RANSAC loop, results in a high level of enhancement. As a consequence of the increasing accuracy of the intrinsic and extrinsic camera parameters the accuracy of camera position is enhanced bringing about a better final outcome.

Chapter 6

Conclusion

Contents

6.1 Conclusion

This master's thesis has proposed an approach for rendering realistic 3D videos by combining a 3D model and video sequences. Our approach requires a pre-computed 3D model which can be reconstructed by utilising modern image-based reconstruction methods. Furthermore, recently evolved 3D reconstruction databases contain an enormous amount of reconstructed models from famous landmarks around the world, which can also be used as input.
First of all, an estimation of the camera position is required; hence, a 6-point DLT algorithm included into a RANSAC framework was implemented. Additionally, a radial distortion model was introduced and optimised by utilising a Levenberg-Marquardt optimiser. Comprehensive evaluations make it clear that including such a radial distortion model significantly enhances the accuracy of the estimated camera parameters and gives rise to more accurate camera positions. Since the 6-point DLT estimates the camera pose and the intrinsic camera parameters, a self-calibration of the camera is implicitly incorporated, resulting in an automatic camera calibration. Hence, our approach has no restrictions regarding camera calibration and 3D modelling; therefore, any standard digital camera can be used to capture the required images and videos, respectively.
The fully automatic approach estimates a depth value for each pre-segmented region, which is extracted using a modern segmentation algorithm. This depth information has been further utilised to render a 3D video or a simple depth map. By using the additional depth information new methods for change detection have been applied to the video sequences resulting in additional information about the scene and the video sequence. As a consequence of this additional information, tasks such as occlusion handling can be simplified. Further experiments evaluating the accuracy of the intrinsic camera parameters have shown the robustness of our implementation.
In conclusion, this master's thesis proposes a novel technique that renders realistic 3D videos assuming a 3D model and a 2D video is provided.

6.2 Future Work

Several ideas for future work are:

- As the experiments in Section 5.3.8 show, it is not always possible to register a video sequence to a 3D model captured at different points in time. Therefore, aligning different 3D models would lead to the possibility of projecting a video to a 3D model captured at a different point in time. This would enable one to visibly show changes between the model and a video sequence with a different time through the back projection of the video onto the 3D model.

- An extension to an indoor environment is possible. The accurate localisation part, in particular, can be used to find the exact position of a camera or a mobile device.

- Semantic tagging can be included. For example, several bounding boxes in one image could tag objects. These bounding boxes could be propagated to every video frame.
- The first version of the program is implemented to run on the CPU and is not runtime optimised. However, the modularity of our approach would allow several parts of the workflow, such as feature detection and 2D to 3D matching, to be run on the GPU to enhance the performance of the program

Appendix A

Acronyms and Symbols

List of Acronyms

1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
AR	augmented reality
BRIEF	binary robust independent elementary features
CCD	charge-coupled device
CMVS	clustering views for multi-view stereo
CPU	Central Processing Unit
DLT	direct linear transform
DOF	degree of freedom
DoG	Difference of Gaussian
Exif	Exchangeable Image File Format
FLANN	fast library for approximate nearest neighbors
GLOH	gradient location and orientation histogram
GPS	Global Positioning System
GPU	Graphics Processing Unit
KNN	k-nearest neighbor
LESH	local energy based shape histogram
LMA	Levenberg–Marquardt algorithm
LoG	Laplacian of Gaussian
MLE	maximum likelihood estimator
PMVS	patch-based multi-view stereo

RANSAC	random sample consensus
RMS	root mean squared
SfM	structure-from-motion
SIFT	Scale-invariant feature transform
SLIC	Simple Linear Iterative Clustering
SURF	speeded up robust feature
SVD	singular value decomposition

Bibliography

[1] Agarwal, S., Furukawa, Y., Snavely, N., Curless, B., Seitz, S., and Szeliski, R. (2010). Reconstructing Rome. *IEEE Computer*, 43(6):40–47.

[2] Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R. (2009). Building Rome in a Day. In *International Conference on Computer Vision (ICCV)*, pages 72–79.

[3] Akbarzadeh, A., Frahm, J., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Merrell, P., Phelps, M., Sinha, S., Talton, B., and Others (2008). Towards Urban 3D Reconstruction from Video. *International Journal of Computer Vision*, 78(2-3):143 – 167.

[4] Arth, C., Klopschitz, M., Reitmayr, G., and Schmalstieg, D. (2011). Real-Time Self-Localization from Panoramic Images on Mobile Devices. In *IEEE International Symposium on Mixed and Augmented Reality*, pages 37–46.

[5] Azizi, N. (2003). Camera Self-Calibration.

[6] Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded Up Robust Features. In *European Conference on Computer Vision (ECCV)*, pages 404–417. Springer.

[7] Bentley, J. L. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517.

[8] Berg, M., Cheong, O., van Kreveld, M., and Overmars, M. (1998). Computational Geometry - Algorithms and Applications. *Annals of Physics*, 54(2):388.

[9] Bernardini, F. and Rushmeier, H. (2002). The 3D Model Acquisition Pipeline. *Computer Graphics Forum*, 21(2):149–172.

[10] Bradski, G. (2000). The OpenCV Library.

[11] Brosch, N., Rhemann, C., and Gelautz, M. (2011). Segmentation-Based Depth Propagation in Videos. In *Austrian Association for Pattern Recognition (OAGM)*.

[12] Brown, D. C. (1971). Close-Range Camera Calibration. *Photogrammetric Engineering*, 37(8):855–866.

[13] Brown, M. and Lowe, D. (2005). Unsupervised 3D Object Recognition and Reconstruction in Unordered Datasets. In *International Conference on 3-D Digital Imaging and Modeling*, pages 56–63.

[14] Bujnak, M., Kukelova, Z., and Pajdla, T. (2011). New efficient solution to the absolute pose problem for camera with unknown focal length and radial distortion. *Asian Conference on Computer Vision (ACCV)*, 6492:8–21.

[15] Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). Brief: Binary Robust Independent Elementary Features. *European Conference on Computer Vision (ECCV)*, 6314:778–792.

[16] Chauve, A., Labatut, P., and Pons, J. (2010). Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1261–1268.

[17] Choi, S., Kim, T., and Yu, W. (2009). Performance Evaluation of RANSAC Family. In *British Machine Vision Conference (BMVC)*, pages 1–12.

[18] Clarke, T. and Fryer, J. (1998). The development of camera calibration methods and models. *The Photogrammetric Record*, 16(91):51–66.

[19] Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *Analysis and Machine Intelligence*, 24(5):603–619.

[20] Crandall, D., Owens, A., Snavely, N., and Huttenlocher, D. (2011). Discrete-continuous optimization for large-scale structure from motion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3001–3008.

[21] Eberhardt, Henning, Klumpp, and Vesa (2010). Density trees for efficient nonlinear state estimation. *Evaluation*, pages 1–8.

[22] Faugeras, O. and Keriven, R. (1998). Complete Dense Stereovision using Level Set Methods. In *European Conference on Computer Vision (ECCV)*, pages 379–393.

[23] Faugeras, O. D., Luong, Q.-T., and Maybank, S. (1992). Camera self-calibration: Theory and experiments. *European Conference on Computer Vision (ECCV)*, 588(1):321–334.

[24] Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, 59(2):167–181.

[25] Fischler, A. and Bolles, C. (1981). Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395.

[26] Fitzgibbon, A. and Zisserman, A. (1998). Automatic camera recovery for closed or open image sequences. In *European Conference on Computer Vision (ECCV)*, pages 311–326.

[27] Fitzgibbon, A. W. (2001). Simultaneous linear estimation of multiple view geometry and lens distortion. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 1:125–132.

[28] Friedman, J. H., Bentley, J. L., and Finkel, R. A. (1977). An Algorithm for Finding Best Matches in Logarithmic Expected Time. *ACM Transactions on Mathematical Software*, 3(3):209–226.

[29] Furukawa, Y., Curless, B., Seitz, S., and Szeliski, R. (2010). Towards internet-scale multi-view stereo. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1434–1441.

[30] Furukawa, Y. and Ponce, J. (2009). Accurate Camera Calibration from Multi-View Stereo and Bundle Adjustment. *International Journal of Computer Vision*, 84(3):257–268.

[31] Furukawa, Y. and Ponce, J. (2010). Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–76.

[32] Gioi, R. G. V., Monasse, P., Morel, J., and Tang, Z. (2011). Self-consistency and universality of camera lens distortion models.

[33] Goesele, M., Snavely, N., Curless, B., and Hoppe, H. (2007). Multi-view stereo for community photo collections. In *International Conference on Computer Vision (ICCV)*, pages 1–8.

[34] Gordon, I. and Lowe, D. (2006). What and where: 3D object recognition with accurate pose. In *Category-level object recognition*, pages 67–82.

[35] Guttmann, M., Wolf, L., and Cohen-Or, D. (2009). Semi-automatic stereo extraction from video footage. In *International Conference on Computer Vision (ICCV)*, pages 136–142.

[36] Haralick, R., Lee, D., Ottenburg, K., and Nolle, M. (1991). Analysis and solutions of the three point perspective pose estimation problem. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 592–598.

[37] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50.

[38] Hartley, R. (1994). Self-calibration from multiple views with a rotating camera. In *European Conference on Computer Vision (ECCV)*, pages 471–478.

[39] Hartley, R. and Zisserman, A. (2006). *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition.

[40] Heikkila, J. (2000). Geometric camera calibration using circular control points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1066–1077.

[41] Hellwich, O. and Sarfraz, M. S. (2005). Head Pose Estimation in Face Recognition across Pose Senarios. In *Energy*, pages 235–242.

[42] Horn, B. K. P. (2003). Tsai's camera calibration method revisited.

[43] Irschara, A., Hoppe, C., and Bischof, H. (2011). Efficient structure from motion with weak position and orientation priors. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*.

[44] Irschara, A., Zach, C., and Bischof, H. (2007). Towards wiki-based dense city modeling. In *International Conference on Computer Vision (ICCV)*, pages 1–8.

[45] Irschara, A., Zach, C., Klopschitz, M., and Bischof, H. (2012). Large-scale, dense city reconstruction from user-contributed photos. *Computer Vision and Image Understanding*, 116(1):2–15.

[46] Irschara, A. and Zach, C. and Frahm, J.-M. and Bischof, H. (2009). From structure-from-motion point clouds to fast location recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2599 – 2606.

[47] Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., and Others (2011a). KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *ACM symposium on User interface software and technology*. ACM.

[48] Izadi, S., Newcombe, R., Kim, D., Hilliges, O., Molyneaux, D., Hodges, S., Kohli, P., Shotton, J., Davison, A., and Fitzgibbon, A. (2011b). KinectFusion: real-time dynamic 3D surface reconstruction and interaction. In *ACM SIGGRAPH 2011 Talks*, page 23.

[49] Jiang, Z.-t., Zheng, B.-n., Wu, M., and Chen, Z.-x. (2009). A 3D Reconstruction Method Based on Images Dense Stereo Matching. In *International Conference on Genetic and Evolutionary Computing*, pages 319–323.

[50] Josephson, K. and Byrod, M. (2009). Pose estimation with radial distortion and unknown focal length. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2419–2426.

[51] Kaminsky, R., Snavely, N., Seitz, S., and Szeliski, R. (2009). Alignment of 3D point clouds to overhead images. In *Conference on Computer Vision and Pattern Recognition Workshops*, pages 63–70.

[52] Kari, M., Rusinkiewicz, S., Ginzton, M., Ginsberg, J., Pulli, K., Koller, D., Anderson, S., Shade, J., Pereira, L., Davis, J., and Others (2000). The digital Michelangelo project: 3D scanning of large statues. In *SIGGRAPH*, pages 131–144.

[53] Kluckner, S., Mauthner, T., Roth, P., and Bischof, H. (2009). Semantic image classification using consistent regions and individual context. In *British Machine Vision Conference (BMVC)*, volume 6, pages 1–7.

[54] Levenberg, K. (1944). A method for the solution of certain problems in least squares. *Quart. Applied Math.*, 2:164–168.

[55] Levinshtein, A., Stere, A., Kutulakos, K. N., Fleet, D. J., Dickinson, S. J., and Siddiqi, K. (2009). TurboPixels: Fast Superpixels Using Geometric Flows. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(12):2290–7.

[56] Li, Y. and Snavely, N. (2010). Location recognition using prioritized feature matching. *European Conference on Computer Vision (ECCV)*, 6312:1–14.

[57] Li, Z., Xie, X., and Liu, X. (2009). An efficient 2D to 3D video conversion method based on skeleton line tracking. In *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video*, pages 1–4.

[58] Longuet-Higgins, H. C. (1981). A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135.

[59] Loop, C. and Zhang, Z. (1999). Computing rectifying homographies for stereo vision. In *Conference on Computer Vision and Pattern Recognition Workshops*, volume 1, pages 125–131.

[60] Lowe, D. (1999). Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV)*, pages 1150–1157.

[61] Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[62] Lucchi, A., Smith, K., Achanta, R., Lepetit, V., and Fua, P. (2010). A fully automated approach to segmentation of irregularly shaped cellular structures in EM images. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 13(2):463–71.

[63] Malisiewicz, T. and Efros, A. (2007). Improving spatial support for objects via multiple segmentations. In *British Machine Vision Conference (BMVC)*. BVMC.

[64] Marquardt, D. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441.

[65] Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proceedings of the Royal Society of London. Series B, Containing papers of a Biological character. Royal Society (Great Britain)*, 207(1167):187–217.

[66] Maybank, S. and Faugeras, O. (1992). A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151.

[67] Mendonca, P. and Cipolla, R. (1999). A simple technique for self-calibration. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 500–505.

[68] Mikolajczyk, K. and Schmid, C. (2005). Performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10):1615–30.

[69] Mori, G. (2005). Guiding model search using segmentation. In *International Conference on Computer Vision (ICCV)*, pages 1417 – 1423.

[70] Moustakas, K., Tzovaras, D., and Strintzis, M. (2005). Stereoscopic video generation based on efficient layered structure and motion estimation from a monoscopic image sequence. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(8):1065–1073.

[71] Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09*, pages 331–340.

[72] Newcombe, R., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). KinectFusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE.

[73] Nistér, D. (2000). Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. In *European Conference on Computer Vision (ECCV)*, pages 649–663.

[74] Nister, D. and Stewenius, H. (2006). Scalable recognition with a vocabulary tree. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2161–2168.

[75] Oskiper, T., Samarasekera, S., Kumar, R., and Sawhney, H. S. (2008). Real-time global localization with a pre-built visual landmark database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[76] Pollefeys, M. and Gool, L. V. (2002). From Images to 3D Models. *Communications of the ACM*, 45(7):50–55.

[77] Pollefeys, M., Koch, R., and Gool, L. V. (1999). Self-Calibration and Metric Reconstruction in spite of Varying and Unknown Internal Camera Parameters. *International Journal of Computer Vision*, 32(1):7–25.

[78] Pollefeys, M., Nistér, D., Frahm, J.-M., Akbarzadeh, a., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S.-J., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewénius, H., Yang, R., Welch, G., and Towles, H. (2007). Detailed Real-Time Urban 3D Reconstruction from Video. *International Journal of Computer Vision*, 78(2-3):143–167.

[79] Pollefeys, M. and Van Gool, L. (1999). Stratified Self-Calibration with the Modulus Constraint. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):707–724.

[80] Pollefeys, M. and Van Gool, L. (2002). Visual modelling: from images to images. *The Journal of Visualization and Computer Animation*, 13(4):199–209.

[81] Raguram, C., Jen, Y., Dunn, E., Clipp, B., Lazebnik, S., and Pollefeys, M. (2010). Building Rome on a Cloudless Day. In *European Conference on Computer Vision (ECCV)*, pages 72–79.

[82] Robertson, D. and Cipolla, R. (2004). An image-based system for urban navigation. In *British Machine Vision Conference (BMVC)*, volume 1, pages 819–828.

[83] Roman, A., Garg, G., and Levoy, M. (2004). Interactive design of multi-perspective images for visualizing urban landscapes. In *Proceedings of the conference on Visualization*, pages 537–544.

[84] Sattler, T., Leibe, B., and Kobbelt, L. (2011). Fast Image-Based Localization using Direct 2D-to-3D Matching. *International Conference on Computer Vision (ICCV)*, 43(7):667–674.

[85] Schaffalitzky, F. and Zisserman, A. (2002). Multi-view matching for unordered image sets, or "How do I organize my holiday snaps?". In *European Conference on Computer Vision (ECCV)*, pages 414–431.

[86] Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1):7–42.

[87] Schindler, G., Brown, M., and Szeliski, R. (2007a). City-Scale Location Recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–7.

[88] Schindler, G., Dellaert, F., and Kang, S. (2007b). Inferring temporal order of images from 3D structure. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–7.

[89] Seitz, S., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 519–528.

[90] Seitz, S. M. (2000). A Theory of Shape by Space Carving. In *International Conference on Computer Vision (ICCV)*, pages 307–315.

[91] Silpa-anan, C. and Hartley, R. (2008). Optimised KD-trees for fast image descriptor matching. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[92] Simon, I., Snavely, N., and Seitz, S. M. (2007). Scene Summarization for Online Image Collections. In *International Conference on Computer Vision (ICCV)*, pages 1–8.

[93] Slabaugh, G., Schafer, R., and Hans, M. (2002). Image-based photo hulls. In *International Symposium on 3D Data Processing Visualization and Transmission*, pages 704–862.

[94] Snavely, N., Seitz, S., and Szeliski, R. (2006). Photo tourism: exploring photo collections in 3D. *ACM Transactions on Graphics (TOG)*, 25(3):835–846.

[95] Snavely, N., Seitz, S. M., and Szeliski, R. (2007). Modeling the World from Internet Photo Collections. *International Journal of Computer Vision*, 80(2):189–210.

[96] Strecha, C., Tuytelaars, T., and Van Gool, L. (2003). Dense matching of multiple wide-baseline views. In *International Conference on Computer Vision (ICCV)*, pages 1194–1201.

[97] Sun, W. (2006). An empirical evaluation of factors influencing camera calibration accuracy using three publicly available techniques. In *Machine Vision and Applications*, volume 17, pages 51–67.

[98] Tavakoli, H. and Pourreza, H. (2010). Automated center of radial distortion estimation, using active targets. In *Asian Conference on Computer Vision (ACCV)*, pages 325–334.

[99] Telcean, F. S. (2007). Master Thesis: Structure from Motion Methods for 3D Modeling of the Ear Canal.

[100] Teller, S., Antone, M., Bodnar, Z., and Bosse, M. (2003). Calibrated, registered images of an extended urban area. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 53(1):93–107.

[101] Tomasi, C. (1992). Shape and Motion from Image Streams under Orthography: A Factorization Method. *International Journal of Computer Vision*, 9(2):137–154.

[102] Triggs, B. (1997). Autocalibration and the absolute quadric. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, page 609.

[103] Triggs, B. (1998). Autocalibration from planar scenes. In *European Conference on Computer Vision (ECCV)*, pages 89–105.

[104] Triggs, B. (1999). Camera pose and calibration from 4 or 5 known 3d points. *International Conference on Computer Vision (ICCV)*, 1:278–284.

[105] Triggs, B., Mclauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (2000). Bundle Adjustment - A Modern Synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, pages 298–372.

[106] Trucco, E. and A. Verri (1998). *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall.

[107] Tsai, R. (1987). A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4):323–344.

[108] Van Meerbergen, G., Vergauwen, M., Pollefeys, M., and Van Gool, L. (2001). A hierarchical stereo algorithm using dynamic programming. In *Proceedings IEEE Workshop on Stereo and MultiBaseline Vision (SMBV)*, pages 166–174.

[109] Varekamp, C. and Barenbrug, B. (2007). Improved depth propagation for 2d to 3d video conversion using key-frams. In *European Conference on Visual Media Production*, pages 1–7.

[110] Vedaldi, A. and Fulkerson, B. (2008). VLFeat: An Open and Portable Library of Computer Vision Algorithms.

[111] Vedaldi, A. and Soatto, S. (2008). Quick shift and kernel methods for mode seeking. *European Conference on Computer Vision (ECCV)*, 4:705–718.

[112] Vergauwen, M. and Gool, L. (2006). Web-based 3D Reconstruction Service. *Machine Vision and Applications*, 17(6):411–426.

[113] Wang, J., Shi, F., Zhang, J., and Liu, Y. (2006). A New Calibration Model and Method of Camera Lens Distortion. In *International Conference on Intelligent Robots and Systems*, pages 5713–5718.

[114] Weng, J., Cohen, P., and Herniou, M. (1992). Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on pattern analysis and machine intelligence*, 14(10):965–980.

[115] Wu, C. (2007). SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT).

[116] Wu, C., Er, G., Xie, X., Li, T., and Cao, X. (2008). A novel method for semi-automatic 2D to 3D video conversion. In *3DTV Conference: The True Vision Capture Transmission and Display of 3D Video*, pages 65–68.

[117] Xiao, J., Chen, J., and Yeung, D. (2008). Structuring visual words in 3D for arbitrary-view object localization. In *European Conference on Computer Vision (ECCV)*, pages 725–737.

[118] Yang, Z. and Cao, F. (2006). Linear Six-Point Algorithm of Camera Self-Calibration and 3D Reconstruction from Single-View or Multi-views. In *Proceedings of the International Conference on Intelligent Systems Design and Applications*, pages 390–395.

[119] Yusoff, F., Rahmat, R., Sulaiman, M., Shaharom, M., and Majid, H. (2009). Establishing the Straightness of a Line for Radial Distortion Correction through Conic Fitting. *Journal of Computer Science*, 9(5):286.

[120] Zhang, G., Hua, W., Qin, X., Wong, T.-T., and Bao, H. (2007). Stereoscopic video synthesis from a monocular video. *IEEE transactions on visualization and computer graphics*, 13(4):686–96.

[121] Zhang, W. and Kosecka, J. (2006). Image Based Localization in Urban Environments. In *International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 33–40.

[122] Zhang, Y.-J. (2006). *Advances in image and video segmentation*. IRM Press.

[123] Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334.

[124] Zhang, Z., Deriche, R., Faugeras, O., and Luong, Q.-T. (1995). A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1-2):87–119.