國立臺灣大學電機資訊學院資訊工程學系

博士論文

Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University

Doctoral Dissertation

動態場景光束法平差
Dynamic Scene Bundle Adjustment

竇菲
Dopfer Andreas

指導教授： 莊永裕博士
Advisor: Yung-Yu Chuang, Ph.D.

中華民國 104 年 6 月
June, 2015

ABSTRACT

This work proposes an extension of Bundle Adjustment to dynamic scenes. In the setting of one or multiple cameras moving in a dynamic environment, the camera pose and the 3D positions of static and moving objects are reconstructed from the captured image sequences. An efficient, low-dimensional representation of the scene is introduced, which is based on approximating trajectories by linear combinations of trajectory bases. Our reconstruction approach requires no knowledge about the objects, not even which are moving or static and is, in difference to other approaches, able to deal with incomplete and noisy data. Experimental evaluation in simulation as well as with real data shows its effectiveness in reconstructing dynamic scenes from moving cameras.

Keywords: Bundle Adjustment, Dynamic Scene, Reconstruction, Structure from Motion, multiple moving cameras, Trajectory Bases, Traffic Scene Reconstruction

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Motivation

BUNDLE Adjustment and Structure from Motion (SfM) methods have matured over the last years to a level where accurate 3D reconstructions from image sequences of historical artifacts, buildings and even whole cities have been created [60, 2]. Some of these reconstructions are actively used by hundreds of millions of users monthly [29]. Bundle adjustment techniques are commonly used as the last step in SfM pipelines, as a refinement stage in real-time Simultaneous Localization and Mapping (SLAM) systems and they have helped create large scale 3D city maps by Google, Apple, Nokia and others. As impressive as these 3D models are, they all have in common that they only consider static environments. Moving objects are filtered out.

The real world we live in is dynamic, everything we do involves some form of motion. We spend a substantial amount of our lifetime in cars, on public transportation or walking. While it is certainly not necessary, and currently not possible, to reconstruct every movement made, there are many scenarios where the ability to reconstruct the dynamic world from

images will be highly useful. Therefore the scenes we are interested in are scenes of our daily lives: busy roads filled with moving cars and motorbikes or intersections with douzens pedestrians walking just to name a few.

One or more of the moving objects have cameras and observe the scene. We introduce a low-dimensional representation for such dynamic scenes based on trajectory bases. Each moving point is seen as a trajectory, and that trajectory is approximated by a linear combination of basis trajectories. This significantly reduces the scenes dimensionality and makes the reconstruction traceable. If knowledge whether a point is static is available it be directly encoded in the proposed representation. Used in a Bundle Adjustment framework our representation has demonstrated the ability to reconstruct challenging real world scenes.

One of the possible scenarios to benefit from our approach is road traffic. Traffic accidents caused an estimated 1.24 million deaths in 2010 [74], while leaving at least 20 times as many people injured. These numbers make traffic accidents the number one cause of death for people aged 15-29. The total number of accidents without people injured is even higher. In 2013 around 3,300 persons got killed on streets in Germany (82 million inhabitants), 290,000 persons got injured and 2.1 million accidents without anyone injured were reported by police authorities [61]. These numbers show the extend of accidents occurring, even in one of the safer countries.

A first step to make traffic safer is to understand why accidents happen. In highly developed countries, data on accidents comes from law enforcement authorities which try to reconstruct what happened in the case of serious accidents. They look at the crash site, the damaged vehicles and question witnesses. It is a time consuming, difficult task so that only a very small fraction of accidents are *reconstructed*. Very few countries have the resources to conduct these reconstructions, and then only accidents with fatalities or severe injuries are reconstructed. Also these reconstructions often miss the wider picture of what happened before the crash. There are big gaps in our knowledge about accidents.

Recently dashboard mounted cameras (dash cams) for cars have become available, and while their use is illegal in a few countries such as Austria and Switzerland), the cameras quickly became ubiquitous in other countries. The reason for this is to have evidence in court in case of accidents, and also to guard against fraud and police corruption. In Russia their use is so common, that when the Chelyabinsk meteor hit the country in 2013 dash cams where the main source for news coverage and the reconstruction of what happened [75]. Since these cameras are widely used, videos of crashes and close calls recorded by these devices flood popular video sharing sites around the world.

Given those available traffic videos, an automated reconstruction method could lead to a deeper understanding of why accidents happen, and what leads to a certain accident. Data about accidents in countries with no reporting system in place can be collected, as well as data about less serious

accidents which are currently seldomly analyzed. Such automated reconstruction of traffic scenes is one of the main usage scenarios of this proposed work. We have discussed the potential impact of these reconstructions in [20], arguing that a framework of several building blocks is required for a comprehensive automated understanding of accidents . Out of these blocks, the ability to estimate the dynamic structure of a scene is the most important, but currently least covered. With the term *dynamic structure* we mean the positions and motion of the recording camera and all other moving entities in the scene.

By completing the proposed work, we will create an important block towards traffic accident analysis. This analysis can potentially decrease accidents by helping to automatically identify dangerous types of traffic situations – so authorities can change existing road layouts to increase safety, or change traffic rules accordingly. Other potential uses are automated traffic rule monitoring so the system could automatically identify illegal or dangerous behavior and allow authorities to enforce laws. Another direct use of such a system is for insurances to verify claims by identifying who or what caused an accident.

Several works have argued that autonomous driving has the potential to make traffic safer [64, 65, 70, 46], since autonomous cars can react faster, do not get tired and will not get distracted. These autonomous vehicles can benefit from automated dynamic scene or traffic understanding by using the information gathered to learn general traffic patterns, learn the behavior of human drivers, or use the data extracted directly in their decision making process.

THESIS STATEMENT

In this chapter, we give an overview about the contributions of the proposed thesis before going into details. The thesis statement is followed by a short description of the significant terms in the statement.

> We propose a novel, practical method to reconstruct dynamic scenes from one or multiple moving cameras. The method extends existing Bundle Adjustment methods to dynamic scenes by modeling the scene and camera motion in an adaptive, low-dimensional space.

Dynamic scenes: A 3D scene in which some or all objects are changing their position over time. Objects follow the rules of physics and therefor follow smooth trajectories. For example, in a street intersection, some cars are moving, people walk on the sidewalk, other cars have temporarily halted or are parked.

Reconstruct: Meaning we aim to find the 3D structure of a scene – we estimate the 3D coordinates of points in the scene, given 2D image observations.

Novel, practical: We want to emphasize that the work we are doing is new – so far bundle adjustment was only used for static scenes. With the term practical we are pointing out that we aim to create a robust method that is able to reconstruct real world scenes from video sequences captured by any kind of camera without the requirement of special hardware or capturing technique.

One or multiple moving cameras: The proposed method can work using a single camera, but also make use of multiple cameras moving independently in the same scene.

Bundle Adjustment: The process of jointly refining camera parameters and 3D scene by minimizing the distance between observed images and the projection of the estimated 3D scene to the image plane. A non-linear least squares optimization problem.

Low-dimensional representation: Reconstructing a dynamic scenes from images of one moving camera is not traceable in its naïve form. A compact, low dimensional representation is required. We represent trajectories of moving objects and moving cameras by a linear combination of predefined bases.

Adaptive representation: The representation adapts, if necessary, to the length of the trajectory, using a larger number of parameters for longer trajectories than for trajectories which are observed only for a short time.

CHAPTER 2

Related Work

THE aim of this thesis is to reconstruct a dynamic scene from a moving camera, which puts our work in the broad category of photogrammetry – the science of making measurements from photographs. This research field is as old as modern photography, dating back to the middle of the nineteenth century. On the other hand, it also fits in the area of computer vision – a field that is besides making measurements, also aims at obtaining a qualitative understanding of images. There are cultural differences between computer vision and photogrammetry, often making the literature on the later hard to access for computer vision researchers. Atkinson and Karara [7, 39] are relatively accessible introductions on non-aerial photogrammetry, [28] is an excellent tutorial paper and [41] is probably the most widely used photogrammetry textbook.

In computer vision, reconstruction of a scene from a sequence of images is SfM – a discipline that has evolved to a mature level. The case of a camera moving through a static scene is solved in a coherent theory [33, 23, 51]. Several systems exist that are robustly able to recover static structure and

camera motion from real world image sequences. The main limitation here is the requirement of a static scene.

The term SfM describes a problem category – reconstructing the structure of a scene given images from a moving camera. One popular method to solve it is based on factorization. Factorization rests on the idea that a measurement matrix containing the image coordinates of P features in F views has a rank of three. This insight was introduced by [66] and [40] independently. The exploitation of this rank three constraint to reconstruct structure and camera motion is known as the Tomasi-Kanade algorithm. In its initial form, it was proposed only for orthographic projection, but later extended to projective models [62, 34]. These extensions require an additional estimation or knowledge of the projective depth for each point which make them often not applicable in practice. Handling of multiple cameras in a factorization-based fashion was introduced by [16]. All the mentioned factorization methods require a full measurement matrix and do not handle outliers well.

The first insight that the human vision system has the ability to reconstruct dynamic structures in SfM like scenarios was found by Johansson in a 1973 study [38]. In the computer vision community, the first attempts to reconstruct dynamic structures began in the 1980s using constraints like rigidity, symmetry or linear representations in low dimensions [69, 12, 58]. With the upcoming of factorization based methods in the 1990s, rigid multi-body factorization was introduced. Multi-body factorization reconstructs scenes consisting of several independently moving rigid objects. For each object, its translation and rotation with respect to the camera are estimated.

One of the first to discuss this approach was [15], since then it has been extended by [24, 47, 57] to handle perspective sequences and recently [54] claimed that multi body factorization has reached a stage that makes it practical useable on realistic sequences. It has several limitations such as a required minimum number of features for each object, and the restriction of working only for rigid objects. None of these limitations will be present in the proposed work.

A more general approach to handle nonrigid structure was introduced by Bregler et al. [11] and termed Nonrigid Structure from Motion (NRSfM). The deformations of a nonrigid object were modeled as a low dimensional set of linear bases, called shape bases - similar to the work in [58]. The shape at each time step is approximated by a weighted sum of K basis shapes. Assuming that K basis shapes can capture the deformation of the scene, Bregler et al.[11] described a rank $3K$ theorem, analogous to the rank 3 theorem in factorization for static scenes. In the nonrigid cases, the shape bases used have to be estimated for each object, since they are specific to the observed data. Initial algorithms lacked the stability of the rigid factorization approach. [77, 10] proposed optimization strategies and extensions to the seminal NRSfM algorithm. Bue [5] suggested to add shape priors, Bartoli et al. [8] used smoothness priors, with the prior being the distance of the deformation from the mean. Others [79, 80, 67] assumed articulated nonrigid objects. The affine camera model for NRSfM was later extended to perspective models [78, 71, 32], however robust solutions able to handle significant nonrigidity remain elusive. Lately Dai et al. [17] introduced an approach using no priors, achieving results similar or superior to prior

based approaches by exploiting the (matrix) structure of the problem in a better way.

Akhter et al. [4] proposed to represent the time varying structure by a collection of trajectories. Their approach is very similar to the shape basis used in previous NRSfM methods, but does not require their calculation for each instance of the problem. The trajectory bases used are precalculated Discrete Cosine Transformation bases. In [3] Akhter et al. showed the duality between both representations. Trajectory based representation for NRSfM have been extended by others since then [50]. Both shape and trajectory based NRSfM approaches are prone to outliers and image noise, additionally they require large camera motions [55].

Another approach on reconstructing 3D structure from images are Active Appearance Models (AAM). This method matches a statistical object model and an appearance model to an observed image. Introduced by Cootes et al. [14], it can match a 3D model to a single image. Using a sequence of images, this method can be used to reconstruct the 3D motion of nonrigid objects such as faces. We have extended the method to be able to benefit from sensor data other than images [21]. Since the method requires a predefined 3D model and a corresponding statistical appearance model learned from observations, it is not the optimal choice for handling dynamic scenes where the appearance of objects changes from scene to scene.

In robotics, Simultaneous Localization and Mapping (SLAM) aims to localize a robot in a map while building the map at the same time [59]. Often SLAM utilizes filter based methods, for which a kalman filter maintains the location of static features and the state of the robot. Techniques based

on monocular cameras have been demonstrated using an extended Kalman filter framework [19, 44]. These monocular SLAM approaches have been improved by inverse depth parametrization [13]. Wang et al. [73] pointed out that SLAM can fail in dynamic situations, if moving objects are not properly dealt with [72, 30]. Attempts to adapt SLAM to dynamic environments have been made. A theoretical framework for simultaneous localization, mapping and moving object tracking (SLAMMOT) was proposed and demonstrated based on LIDAR data Wang et al. [73]. An approach to enable monocular SLAM for dynamic scenes [48, 35] augments static and moving features into the state vector. Bearing-only tracking and monocular SLAM are solved concurrently. The significant difference of all SLAM approaches to the proposed work is that they aim to reconstruct the scene incrementally, while the proposed method looks at the complete data.

Reconstruction of scenes, no matter what method is being used, is prone to noise and outliers. Therefore an optimization stage can be added in which the reprojection error is being minimized. This technique is called Bundle Adjustment (BA). In some forms, this method was already used in early photogrammetry methods, even before computers were around. System implementations using factorization or filtering based approaches often use BA as the last stage in their pipeline, or as a method that is applied in intervals to enhance consistency. BA is also used as the primary way of reconstruction, with factorization or triangulation only needed to generate reasonable starting points. An excellent overview over BA techniques is given by Triggs et al. [68]. Thanks to increased computing power, and even more due to improvements in the optimization stack used, large scale BA problems – up to city scale – have become solvable [37, 1, 76].

Several attempts have been made to attack the reconstruction of dynamic scenes, based on data from multiple cameras, triangulation after establishing the camera poses, or a combination of them. Park [55] reconstructed dynamic scenes from images taken by multiple cameras (not video sequences), but the camera poses were calculated in a separate step using known static background. Zou and Tan [82] introduced a SLAM based method using multiple moving cameras capable of reconstructing the environment and tracking moving objects. Kundu et al. [42] suggested an incremental monocular SLAM, that can either track moving objects and reconstruct the static scene only, or also reconstruct moving objects given they are observed in a proper way. [82, 42] evaluated their algorithms only on sequences with a very low number of objects. This makes their works not comparable to ours, also their work is a framework built on a sequence of different processing steps. Geiger [25] aimed to understand traffic scenes based on short video sequences by introducing a probabilistic framework combining car detection and lane detection, occupancy grid maps, scene flow and vehicle tracklets achieving remarkable performance. His work is, in contrast to ours, based on stereo camera sequences.

Even though our work does not resemble any of the here discussed work directly, it has similarities with some of them. Multi-body factorization aims to reconstruct the 3D motion of objects – and while it reconstructs the motion together with the objects orientation it requires sufficient features for each object. Our work in contrast aims to reconstruct the trajectories of single feature points.

NRSfM methods reconstruct the 3D motion of single features like we do and can be seen as an approach able to solve similar scenarios. That is why we will compare the performance of our work to NRSfM methods. NRSfM, as well as multi-body factorization, is based on factorization which makes them not as robust to image noise, missing data and wrong data associations as the proposed method. Our work utilizes robust statistical methods to deal with real world scenarios. Also the used framework allows direct integration of additional sensor data. As the title of this thesis already suggests, bundle adjustment methods are similar to our work. Our work is an extension to BA allowing it to deal with dynamic scenes – which has not been done to date.

CHAPTER 3

Background

THIS chapter aims to introduce several concepts and methods used. They are the building blocks and the background this work is based on, and help understand the concepts later introduced. First camera projection models will be discussed in Section 3.1. This is necessary since this thesis circles around the concept of reconstructing 3D data given 2D image observations, which is essentially the inversion of what cameras do: project 3D points to a 2D image plane. After describing affine and projective models, image features will be shortly discussed. The proposed work is based on point features rather than image intensity values, so they need to be characterized as well as their noise characteristics and possible error cases, which is done in Section 3.2.

How to represent 3D scenes will be elaborated in Section 3.3. Starting from static scenes representations for dynamic scenes, scenes in which some or all points change their position over time, are discussed. After that Bundle Adjustment and with it different optimization techniques that are used for it is explained. Loss functions that are used to make the method more

robust are introduced. They are specially important when dealing with imperfect real world data.

3.1. Camera Models

This thesis is about reconstructing real world scenes given images, or in other words how to get 3D coordinates from a set of 2D image observations. Therefore we start by discussing how the 2D image observations were created: the projection models of cameras.

Most real world cameras are *projective* cameras. They can be represented by a pinhole camera with additional terms for distortion. The *affine* or parallel projection camera model is a simpler model in which the camera is only represented by its orientation and the projection is parallel. It can be seen as a special case of the perspective model (image point at infinity). This model allows convenient manipulation - in proper matrix form one matrix multiplication can reduce a 3D scene to its image. We use this model as a proof of concept due to its simplicity and to compare our work to NRSfM techniques which almost exclusively use this model.

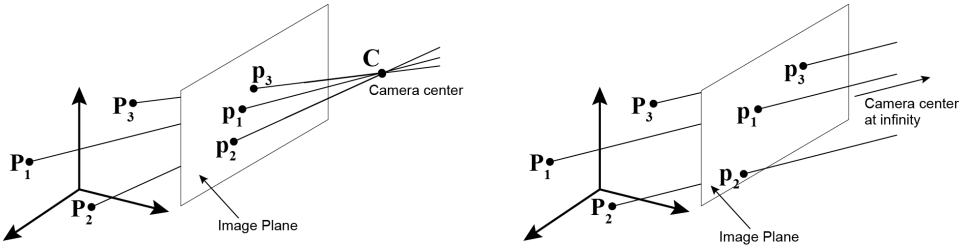


FIGURE 3.1. Schematic comparison between perspective and affine projection models. The left image shows perspective projection, the right image shows affine projection.

Figure 3.1 gives an overview of the basic difference between the two models discussed. We are aware that a multitude of projection models exist, but we restrict ourself to the two that are relevant for our work. The notations used follow the book from Zisserman and Hartley [33].

A point $\mathbf{P}$ that is defined by the orthogonal coordinates $\mathbf{P} = (X, Y, Z)$ is projected to the point $\mathbf{p} = (x, y)$ on the image plane by a camera. An *affine* camera is defined by its orientation, given as a rotation matrix $\mathbf{R}$ truncated to 3×2. The affine projection can be found by

$$\mathbf{p} = \mathbf{P} \cdot \mathbf{R}$$

Projective cameras are defined by a focal length f and its pose which consists of camera rotation $\mathbf{R}$ and its translation $\mathbf{t}$. The pose will change for each frame in the case of an image sequence taken by a moving camera. A 3D point $\mathbf{P}$ is projected onto the camera image $\mathbf{p} = (x, y)$ by

$$(X, Y, Z)^T \mapsto (f \cdot X/Z, f \cdot Y/Z)^T.$$

Taking into account that the origin of coordinates in the image plane is not necessarily at the principal point p and expressing the equation it in homogenous coordinates we get

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f \cdot X/Z + p_x \\ f \cdot Y/Z + p_y \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.1}$$

The matrix to project points to the image is called the camera calibration matrix K.

$$K = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

Equation 3.1 can then be written in compact form

$$\mathbf{p} = K[I \mid 0]\mathbf{P},$$

taking into account the camera pose we get

$$\mathbf{p} = K[\mathbf{R} \mid \mathbf{t}]\mathbf{P}.$$

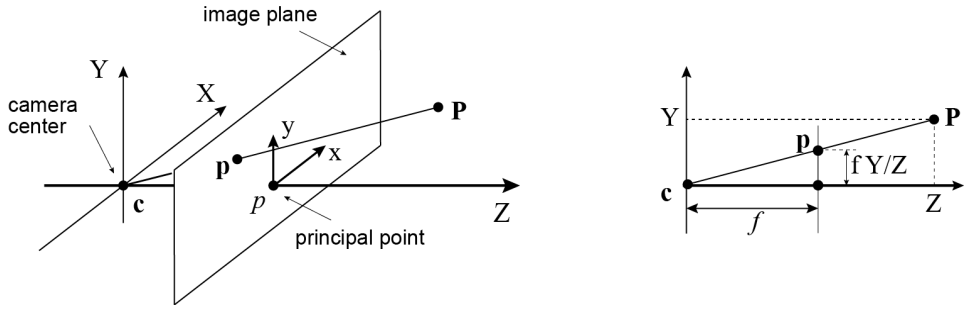


FIGURE 3.2. Details on perspective projection. The left image defines camera and image centric coordinate systems. the right image shows the perspective projection and defines the focal length.

Figure 3.2 gives an overview over the camera parameters for perspective projection. As radial distortion is present to some degree in most real world cameras, we model it with two radial distortion parameters k_1, k_2. Although different distortion models exist, we focus on a two parameter one here.

$r(\mathbf{p})$ is a function for radial distortion and $||\mathbf{p}||$ is the distance to the image center.

$$||\mathbf{p}|| = \sqrt{(\mathbf{p}_x - p_x)^2 + (\mathbf{p}_y - p_y)^2},$$
$$r(\mathbf{p}) = 1.0 + k_1 * ||\mathbf{p}||^2 + k_2 * ||\mathbf{p}||^4.$$

This gives a projection in pixels, where the origin of the image is the center of the image, the positive x-axis points right, and the positive y-axis points up. In the camera coordinate system, the positive z-axis points backwards, so the camera is looking down the negative z-axis.

As a short summary we list the steps necessary to project a 3D point $\mathbf{P}$ onto an image based on the camera parameters $f, \mathbf{R}, \mathbf{t}, k_1, k_2$

$$\mathbf{P}' = \mathbf{R} \cdot \mathbf{P} + \mathbf{t}$$
$$\mathbf{p}' = -\mathbf{P}'/\mathbf{P}'_z$$
$$\mathbf{p} = f \cdot r(\mathbf{p}') \cdot \mathbf{p}'$$

The first line converts from world to camera coordinates, the second line is the actual perspective division where $\mathbf{P}_z$ is the projective depth. The third line is the conversion to image coordinates. Summarizing the perspective projection we get

$$\mathbf{p} = proj(\mathbf{P}, \mathbf{C}_{pose}, \mathbf{C}_{int}) \tag{3.2}$$

where $\mathbf{C}_{pose}$ represents the camera pose, consisting of the camera center position and the camera orientation (6 parameters) and $\mathbf{C}_{int}$ stands for the intrinsic camera parameters such as focal length, image center, and distortion (we use 5 parameters).

3.2. Feature Points

This work is based on feature points, therefore their characteristics will be shortly discussed. Image features are represented by their position on the image plane. They are a tuple of coordinates, given either in pixels or after applying the intrinsic camera parameters as metric distances. An image feature in the context of this work is a point on images that represents a real world 3D point. In subsequent images, the image feature shall denote the same real world 3D point even when the camera sees the point from different directions or distances.

Numerous kinds of feature point detectors exist, each with different characteristics. Some methods such as SIFT [49] or HoG[18] use an area of the image to characterize the feature point and identify the same point in other images. They are robust to scaling and rotations. Other simple detectors such as Harris [31] corner detector rely on optical flow to associate simple features in subsequent images. A good overview of the state of the art in feature point detection and tracking is given in the computer vision textbook of Szelinski [63]. In this work we use corner features that are tracked using optical flow for single camera scenarios. When using multiple cameras, different or additional methods might be used to ensure correct data association between different camera sequences. Since these feature points

are sparse, the work at hand represents a sparse 3D reconstruction, unlike dense methods such as stereo.

All observations of image features can be stacked into a *measurement matrix* $\mathbf{Z}$, which is a $2P \times F$ matrix with P being the number of features, and F the number of frames in the sequence.

$$\mathbf{Z} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1P} \\ y_{11} & y_{12} & \cdots & y_{1P} \\ x_{21} & x_{22} & \cdots & x_{2P} \\ y_{21} & y_{22} & \cdots & y_{2P} \\ \vdots & & \ddots & \vdots \\ x_{F1} & x_{F2} & \cdots & x_{FP} \\ y_{F1} & y_{F2} & \cdots & y_{FP} \end{bmatrix}$$

3.2.1. Noise and Errors

Since a digital image is a discrete array of intensities, the features found in an image will be tainted by discretization noise of some form. Some feature detectors resolve features to sub pixel resolution. This may reduce the noise, but not eliminate it. Higher resolution image sequences may also allow an reduction of this error. Features detected and tracked over a sequence of frames can drift, also leading to errors.

Another form of error is when a feature is not observed by the camera anymore because of whatever reason. This will lead to an incomplete measurement matrix. Worse than not observing a feature anymore is confusing two features. This is called incorrect data association (DA) and can lead to serious errors. Many reconstruction methods, especially factorization-based ones, tend to fail completely in presence of wrong DA. Our approach

is able to handle both missing data as well as wrong DA to some extent. More details will be discussed in the remainder of this thesis.

3.3. Scene Representation

First the representation of static scenes is discussed, from which possible representations of dynamic scenes are derived. The 3D structure of the scene is denoted as $\mathbf{S}$, a matrix with the coordinates of one point in each column.

$$\mathbf{S} = \begin{bmatrix} X_1 & \cdots & X_P \\ Y_1 & \cdots & Y_P \\ Z_1 & \cdots & Z_P \end{bmatrix}$$

P is the number of points in the static scene, so the size of $\mathbf{S}$ is $3 \times P$. If we consider a scene with moving objects, $\mathbf{S}$ becomes $\mathbf{S}(t)$ where the t stands for the time at which the scene is represented. The dynamic scene is a sequence of *snapshots* of the scene.

$$\mathbf{S}(f) = \begin{bmatrix} X_1 & & X_P \\ Y_1 & \cdots & Y_P \\ Z_1 & & Z_P \end{bmatrix}; \quad \mathbf{S}_{dyn} = [\mathbf{S}(1)^T\ \mathbf{S}(2)^T \cdots\ \mathbf{S}(F)^T]^T$$

where F is the temporal length of the sequence denoting the total number of frames and f the corresponding index. The size of $\mathbf{S}_{dyn}$ is then $F \times 3P$. We can restructure $\mathbf{S}_{dyn}$ into

$$\mathbf{S}_{dyn^*} = \begin{bmatrix} X_{11} & \cdots & X_{1P} & Y_{11} & \cdots & Y_{1P} & Z_{11} & \cdots & Z_{1P} \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ X_{F1} & \cdots & X_{FP} & Y_{F1} & \cdots & Y_{FP} & Z_{F1} & \cdots & Z_{FP} \end{bmatrix}$$

to come to a useful form. Another way of representing the dynamic scene is to acknowledge that the set of changing positions of each moving point represents a trajectory. The set of all these trajectories then represents the dynamic scene.

$$\mathbf{t}(p) = \begin{bmatrix} X_1 & & X_F \\ Y_1 & \cdots & Y_F \\ Z_1 & & Z_F \end{bmatrix}; \quad \mathbf{S}_{dyn} = [\mathbf{t}(1)^T \mathbf{t}(2)^T \cdots \mathbf{t}(P)^T]^T \tag{3.3}$$

In addition to the scene itself, the extrinsic and intrinsic parameters of the cameras are needed as described in Section 3.1. All parameters being estimated, such as the scene structure, camera focal length, distortion and poses will be parameterized in a single large state vector $\mathbf{X}$.

3.4. Nonlinear Optimization for Bundle Adjustment

Bundle adjustment is the problem of refining a visual reconstruction to *jointly* produce *optimal* 3D structure and viewing parameter estimates (camera pose, orientation and camera parameters). Optimal means that the parameter estimates are found by minimization of some cost function that quantifies the model fitting error. Jointly states that the solution is simultaneously optimal with respect to both structure and camera.

Bundle adjustment is a large, sparse geometric parameter estimation problem, with the parameters being the combined 3D feature coordinates, camera poses and calibrations. Traditionally these adjustment computations are formulated as nonlinear least squares problems. The cost function being quadratic in feature reproduction error with explicit outlier handling. Newer approaches often use non-quadratic M-estimator like distributional

models to integrate outlier handling, and also potentially include additional penalties for overfitting [68].

This section gives an overview over the state of the art BA techniques that are used for reconstruction. The core of the estimation is a nonlinear least squares problem minimizing the distance between the observed image and the reprojection of the estimated 3D scene to the image plane. Our work makes use of these optimization techniques. More elaborate discussions of nonlinear optimization and BA techniques are given by Triggs et al.[68] and Nocedal and Wright [53].

3.4.1. Nonlinear Least Squares

The vector $x \in \mathbb{R}$ is a n-dimensional vector of variables and $F(x) = [f_1(x), \cdots, f_m(x)]^T$ is a m-dimensional function of x. To put it into the context of what was previously discussed: x is the state vector $\mathbf{X}$ that consists of all variables to estimate. $F(x)$ is the error function with the error being the 2D distance between observed image feature position, and the position calculated using the state vector and the projection function. It might further contain a loss function (will be discussed in detail in Section 3.4.3). We aim to solve

$$\begin{aligned} \arg\min_x \frac{1}{2}\|F(x)\|^2 . \\ L \leq x \leq U \end{aligned}$$

where L and U are upper and lower bounds on x. A global minimization of $F(x)$ is in general not traceable, therefore the aim is to find a solution by

solving a sequence of approximations instead of solving the original problem.

$J(x)$ is the Jacobian of $F(x)$ ($J_{ij}(x) = \partial_j f_i(x)$, $m \times n$ matrix). The gradient vector $g(x) = \nabla \frac{1}{2}\|F(x)\|^2 = J(x)^T F(x)$. An approximation can be constructed by using the linearization $F(x + \Delta x) \approx F(x) + J(x)\Delta x$ which leads to

$$\min_{\Delta x} \frac{1}{2}\|J(x)\Delta x + F(x)\|^2$$

Depending on how the step size is evaluated, two major categories of optimization algorithms emerge: *trust region* and *line search* methods. In some sense the two methods are dual to each other. Trust region methods first choose a step size (the size of the trust region) and then a step direction, while line search methods first choose a step direction and then the size of the step [9].

In the following sections we will discuss some of the possible algorithms in more detail.

3.4.2. Trust Region Methods

Trust Region algorithms approximate the objective function using a model function (often quadratic) over a subset of the search space [9]. This search space is called the trust region. If the model function minimizes the true objective function, then the trust region is expanded. Otherwise the trust region is contracted and the model optimization problem is solved again. The basic trust region algorithm is listed in Algorithm 1.

Algorithm 1 Basic Trust Region Algorithm

1: Start from initial guess x and a trust region radius μ.
2: Solve:

$$\begin{aligned} \arg\min_{\Delta x} \frac{1}{2}\|J(x)\Delta x + F(x)\|^2 \\ \text{such that } \|D(x)\Delta x\|^2 \leq \mu \\ \text{and } L \leq x + \Delta x \leq U. \end{aligned}$$

3: calculate

$$\rho = \frac{\|F(x+\Delta x)\|^2 - \|F(x)\|^2}{\|J(x)\Delta x + F(x)\|^2 - \|F(x)\|^2}.$$

4: if $\rho > \epsilon$ then $x = x + \Delta x$.
5: if $\rho > \eta_1$ then $\rho = 2\rho$.
6: else if $\rho < \eta_2$ then $\rho = \frac{\rho}{2}$.
7: GOTO 2.

$D(x)$ is a matrix defining a metric on the domain of $F(x)$, ρ is a measure for the quality of the step size Δx. It measures how well the linear model predicts the decrease in the value of the non-linear objective. The increase or decrease of the trust region radius depends on this measure ρ.

The key computational step in the algorithm is the solution of a constrained optimization problem (line 2 in Algorithm 1). There are different ways of solving this problem, each giving rise to a different trust-region algorithm. In the following, we will describe two of them: Levenberg-Marquardt and Powell's Method (also called dogleg method).

Levenberg-Marquardt. The Levenberg-Marquardt algorithm [45, 52] was the first trust-region algorithm developed, and is still one of the most popular methods to solve non-linear least squares problems to date.

The solution to line 2 in Algorithm 1 can be obtained by solving an unconstrained optimization problem of the form

$$\arg\min_{\Delta x} \frac{1}{2}\|J(x)\Delta x + F(x)\|^2 + \lambda\|D(x)\Delta x\|^2$$

λ being a Lagrange multiplier that is inversely related to μ. Let $D(x)$ be a non-negative diagonal matrix (typically the square root of the diagonal of $J(x)^T J(x)$).

$$\arg\min_{\Delta x} \frac{1}{2}\|J(x)\Delta x + F(x)\|^2 + \frac{1}{\mu}\|D(x)\Delta x\|^2 \tag{3.4}$$

After concatenating the matrix $\sqrt{\mu}D$ to the bottom of J and adding zeros to the vector of f, we get the following simpler form

$$\min_{\Delta x} \frac{1}{2}\|J(x)\Delta x + f(x)\|^2.$$

This equation dominates the computational cost of the algorithm in most cases. There are two ways of solving it: either by factorization (exact step Levenberg - Marquardt algorithm), or iterative (Inexact step Levenberg - Marquardt algorithm). Factorization based methods use a Cholesky or a QR factorization to compute an exact solution of Equation 3.4. Inexact solutions are based on truncated Newton methods [53]. The used implementation of the proposed approach is based on Levenberg Marquardt. It is using sparse Cholesky factorization.

3.4.3. Loss Functions

For least squares problems where input data might contain outliers, it is important to use a loss function, a function that reduces the influence of these outliers. Otherwise a few outliers can easily drag the solution away from the correct value. A robust loss function reduces the error for outliers, leading them to have a lower weight and not overly influence the final solution.

$$\begin{aligned} \min_{\mathbf{x}} \quad & \frac{1}{2}\sum_{i}\rho_i\left(\|f_i\left(x_{i_1},\cdots,x_{i_k}\right)\|^2\right). \\ & L \leq x \leq U \end{aligned}$$

$\rho_i(\cdot)$ is a *loss function*. It is a scalar valued function that has the purpose of reducing the influence of outliers on the solution of the non-linear least square problem. When we set the loss function as identity function $p_i(x) = x$, and the bounds to $L = -\infty$ and $U = \infty$, we get the familiar classical unconstrained non-linear least squares problem. The loss functions used are scalar valued functions. In robust statistics they are called M-estimators (from Maximum likelihood-type). Below some possible functions and their respective curves. In our formulation s is the squared error.

Quadratic:

$$\rho(s) = s$$

Huber type:

$$\rho(s) = \begin{cases} s & s \leq 1 \\ 2\sqrt{s} - 1 & s > 1 \end{cases}$$

Cauchy type:

$$\rho(s) = \log(1 + s)$$

Arctan type:

$$\rho(s) = \arctan(s)$$

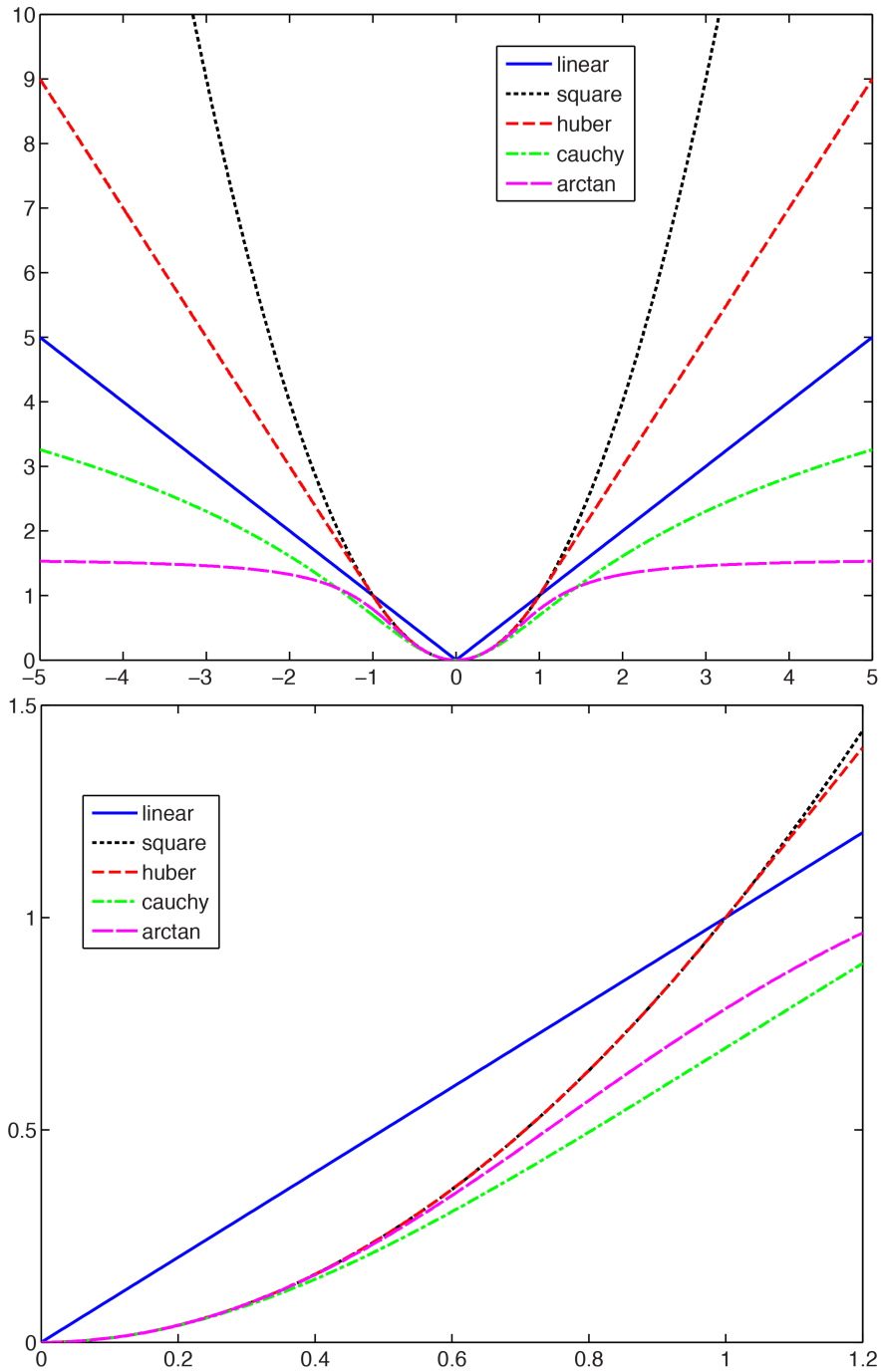


FIGURE 3.3. Top: Shape of different loss functions. Bottom: detailed view for the range [0 1.2]

CHAPTER 4

Dynamic Scene Representation

In this chapter the core contributions of this paper are presented - compact, efficient representations for dynamic scenes. First the dimensionality of the problem will be analyzed to illustrate the need for a compact representation. Then the proposed representation is introduced, its implications are elaborated and its representational power is verified. The case of a full measurement matrix Z is explained first and then expanded to the scenario of an incomplete measurement matrix. Also the representation of camera poses, how static objects are treated, and how additional priors can be included is discussed. After that ways to find reasonable starting conditions for the bundle adjustment step are introduced, and requirements for reconstructability in this problem category are elaborated. We look at the case of reconstruction from a single camera first, then extend to multiple cameras and elaborate on the benefits .

4.1. Fundamentals

The problem we aim to solve is underdetermined, meaning there are more unknowns than equations. We first assume that the measurement matrix **Z** is full, meaning that in every frame every feature is observed. Figure 4.1 illustrates the difference between a fully and a partially observed scene.

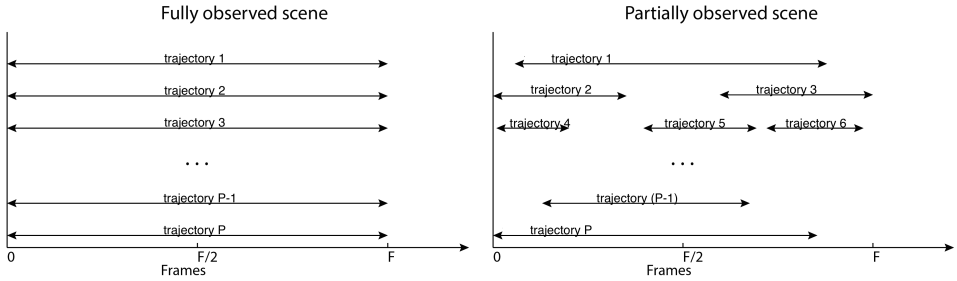


FIGURE 4.1. Scene observations. In the left image every trajectory is observed for every frame of the scene. In the right image features are not visible all the time. Multiple gaps can occur leading to several trajectories belonging to the same physical object.

Given that the number of frames is F and the number of trajectories is P, we have $F \times 2P$ measurements. In a static environment we are looking for $3P + 6F$ parameters – three components for the position of each point, and six camera parameters per frame. These six camera parameters separate into three translative and three rotative components since the camera is moving. Given a reasonable number of frames and points this system is well defined (typically F and P are large, so $(F \times 2P) > (3P + 6F)$). For a short video sequence of 5 seconds we assume to have 150 frames (30 fps, $F = 150$) and 20 points. These numbers are just picked to illustrate the discussed dimensionality. The numbers at hand give us 6000 observations for $60 + 900$ unknown parameters making the problem generally traceable.

In a dynamic environment the problem becomes more challenging. While the number of camera related parameters stays the same, we need $3 \times F$ parameters instead of only 3 for each moving object. This leaves us with a more problematic ratio between measurements and parameters to estimate (typically $(F \times 2P) < (F \times 3P + 6F)$). We make the assumption that every point is moving. Given the numbers used before for illustration we still have 6000 observations, but now $9000 + 900$ unknown parameters – clearly an intractable scenario.

4.2. Trajectory Bases

This Section describes the concept of utilizing Trajectory Bases to compactly represent trajectories. Possible choices of bases are discussed and compared, the representational power of the chosen bases analyzed and the effect of the number of Bases (K) elaborated.

4.2.1. Concept

The moving objects we aim to reconstruct are real world objects, which means that the speed and acceleration of each object is bounded. This is a reasonable assumption given that the moving objects need to obey the laws of physics. This assumption leads to another insight: the trajectory of each object is smooth. Jumps in position of an object would require indefinite acceleration, which is not possible for real world objects.

To represent the trajectory of each moving object, we use a linear combination of K predefined trajectory bases. $\theta^j \in \mathbb{R}^F$ is a trajectory basis vector of length F and $a_{xj}(i)$, $a_{yj}(i)$ and $a_{zj}(i)$ are the coefficients corresponding to

that basis vector. The form stated in Equation 3.3 can be approximated as follows using trajectory bases:

$$\mathbf{t} = \begin{bmatrix} X_1 & & X_F \\ Y_1 & \cdots & Y_F \\ Z_1 & & Z_F \end{bmatrix} \quad \rightarrow \quad \mathbf{t} = \begin{bmatrix} \theta^1 \cdot a_x^1 + & \cdots & + \theta^K \cdot a_x^K \\ \theta^1 \cdot a_y^1 + & \cdots & + \theta^K \cdot a_y^K \\ \theta^1 \cdot a_z^1 + & \cdots & + \theta^K \cdot a_z^K \end{bmatrix} \tag{4.1}$$

Each trajectory is approximated using $3K$ trajectory coefficients (trajectory parameters), K coefficients for each dimension. Figure 4.2 illustrates the concept and Figure 4.3 shows the first 10 bases of the used representation. The choice of bases is discussed in Section 4.2.2.

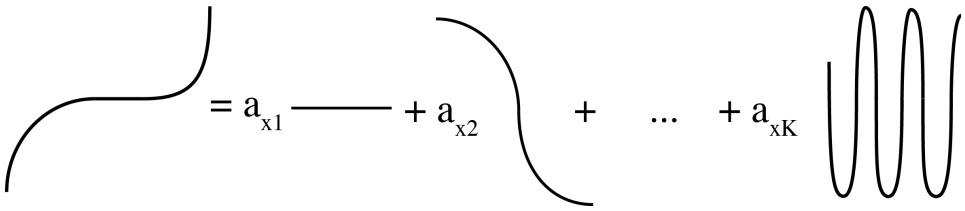


FIGURE 4.2. Trajectory bases representation. The trajectory on the left is approximated by a linear combination of trajectory bases

A compact form for the whole scene can be obtained using the previously defined notation for $\mathbf{S}_{dyn^*}$. Structuring the trajectory coefficients a in a matrix $\mathbf{A}$ and doing similar with the basis vectors θ into $\mathbf{\Theta}$ we can obtain the following form

$$\mathbf{S}_{dyn^*} = \mathbf{\Theta}\mathbf{A} \tag{4.2}$$

with

$$\mathbf{A} = [A_x \; A_y \; A_z];$$

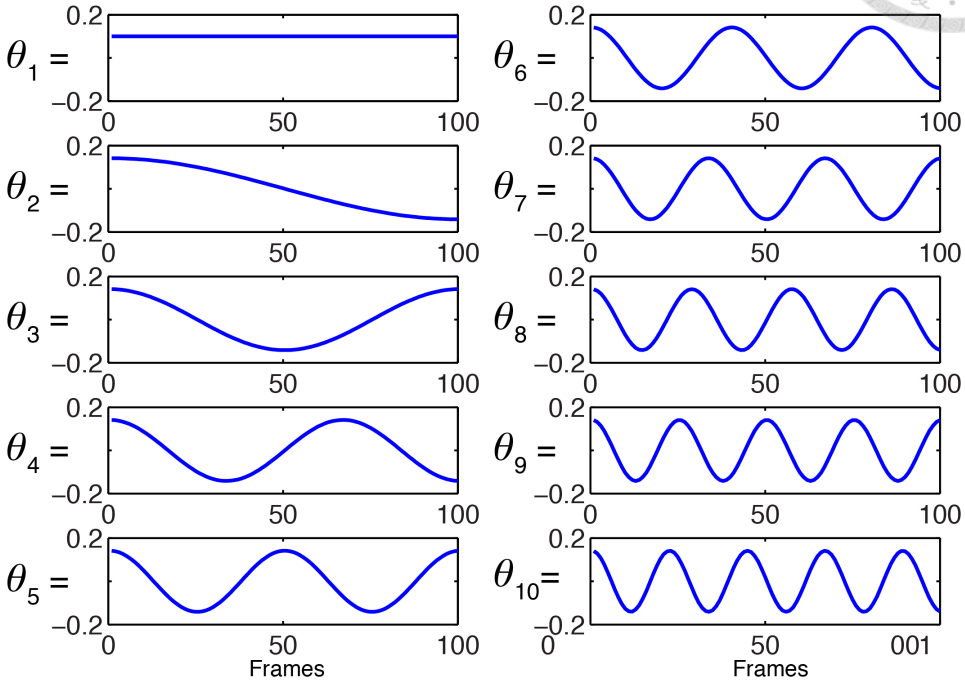


FIGURE 4.3. First 10 bases of the trajectory bases used.

$$A_x = \begin{bmatrix} a_x^1(1) & \cdots & a_x^1(P) \\ \vdots & & \vdots \\ a_x^K(1) & \cdots & a_x^K(P) \end{bmatrix};$$

$$A_y = \begin{bmatrix} a_y^1(1) & \cdots & a_y^1(P) \\ \vdots & & \vdots \\ a_y^K(1) & \cdots & a_y^K(P) \end{bmatrix};$$

$$A_z = \begin{bmatrix} a_z^1(1) & \cdots & a_z^1(P) \\ \vdots & & \vdots \\ a_z^K(1) & \cdots & a_z^K(P) \end{bmatrix};$$

$$\mathbf{\Theta} = \begin{bmatrix} \theta_1^T & & & \\ & \theta_1^T & & \\ & & \theta_1^T & \\ & \vdots & & \\ \theta_P^T & & & \\ & \theta_P^T & & \\ & & \theta_P^T & \end{bmatrix};$$

This way we represent the dynamic 3D structure of a scene using a *small number* of trajectory coefficients $\mathbf{A}$, making the problem of bundle adjustment for dynamic scenes traceable. Originally the number of parameters to estimate in $\mathbf{S}_{dyn}$ was $(3P \times F)$, using trajectory basis representation it is reduced to $(3P \times K)$. Typically $K \ll F$. Using the same numbers like before ($F = 150,\ P = 20$) and a K of 12 we get 6000 observations and $720 + 900$ unknowns. This makes the problem traceable from a mathematical viewpoint. K is typically in the range of $6 - 20$, more about its influence is discussed in the next section. Later we discuss the possibility of further reducing the dimensionality by representing the camera pose in a similar fashion. The apparent reduction in dimensionality results from the fact that the described representation implies smooth trajectories.

The idea of representing a trajectory, or more generally speaking any discrete signal, by a linear combinations of bases is not new. Lossy audio and image compression use DCT to compress their data, and Akhter et al. [4, 3] use it to represent nonrigid structure in NRSfM. To the best of our knowledge we are the first to introduce it to the context of bundle adjustment for dynamic scenes.

4.2.2. Choice of Trajectory Bases

Trajectory bases are sets of finite discrete data points used to express the trajectories of moving objects. Discrete Fourier Transform (DFT), Discrete Sine Transform (DST) and Discrete Cosine Transform (DCT) [56, 36] use sums of sinoid functions of different frequencies to represent a signal. The three methods are closely related. They distinguish itself by the use of either sine/cosine functions or both - and weather they operate on real numbers only rather than complex ones.

The periodicity of DFTs leads to symmetrical boundaries which is not desirable for the application at hand, and similar problems arise for DSTs. Therefore DCTs are used as trajectory bases in this work.

For static scene Bundle Adjustment all reconstructed 3D points are independent of each other, and only depend on the camera pose at the time when they are observed. When reconstructing trajectories of moving objects the resulting Jacobian matrix in the optimization step is sparse. A sparse Jacobian matrix allows faster computation then with a full matrix. By using trajectory bases every parameter used to represent a trajectory influences every point in the trajectory, leading to a non-sparse block in the Jacobian matrix for each trajectory.

A choice of basis functions that are zero over wide areas of its length will lead to a more sparse Jacobian. Splitting the used DCT bases into smaller chunks and setting them to zero outside is a possibility, as well as creating a function that piecewise defines the function. Figure 4.4 shows DCT trajectory bases split into 3 regions. Figure 4.5 shows a piecewise defined basis

set, where one trajectory parameter only influences a small group of points, with small overlap for points at the edge of one base.

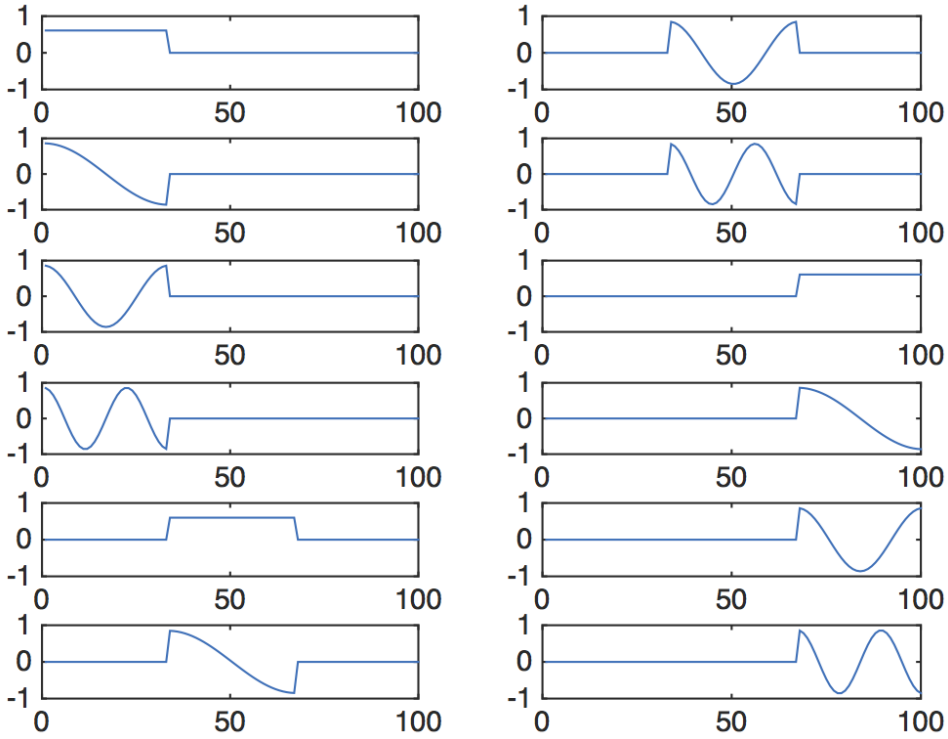


FIGURE 4.4. Split DCT bases. Instead of 12 Bases the sequence is split in 3 parts and each using 4 bases for representation. This way the first third of the trajectory factors will only influence the first part of the trajectory, the next third the next third of the trajectory, and so on.

In Figure 4.6 the effect of these choices of basis functions can be seen. Increasing zero sections in the bases increase the sparsity of the Jacobian matrix. As a tradeoff splitting the trajectory introduces discontinuities at the boundaries. The accuracy with which a trajectory can be represented decreases due to this discontinuities.

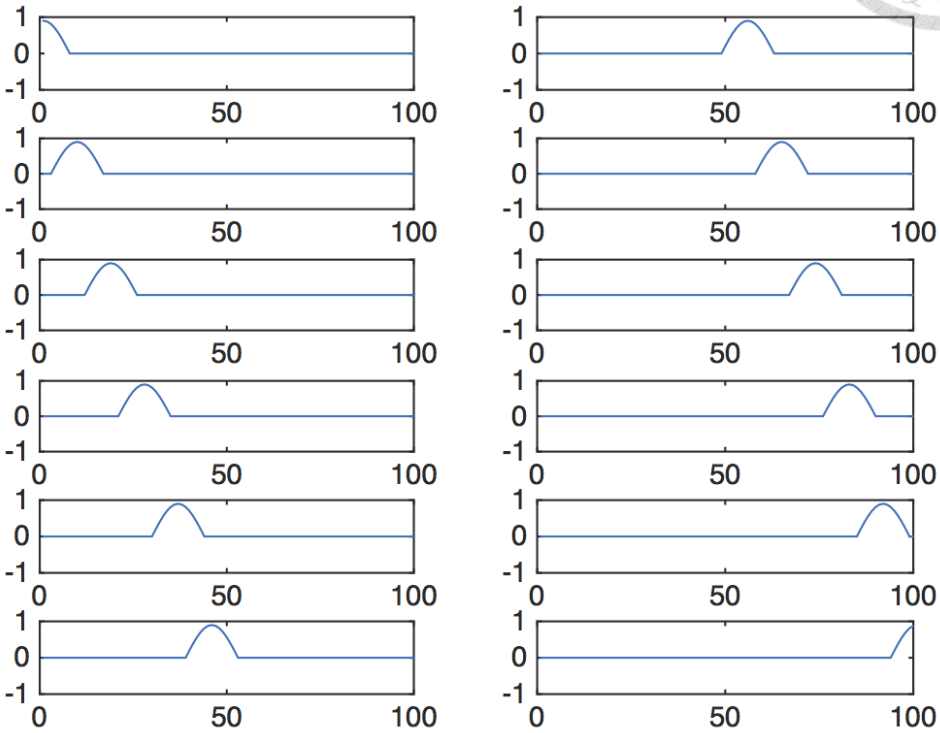


FIGURE 4.5. Piecewise defined bases. Each trajcetory factor will only influence a small portion of the trajectory resulting in a sparse Jacobian martix.

Concluding we can see that a sparser Jacobian would lead to faster computation, but lower accuracy. Since our method is an offline batch-processing type of calculation we weight accuracy over speed and use full DCT trajectory bases thru out this work.

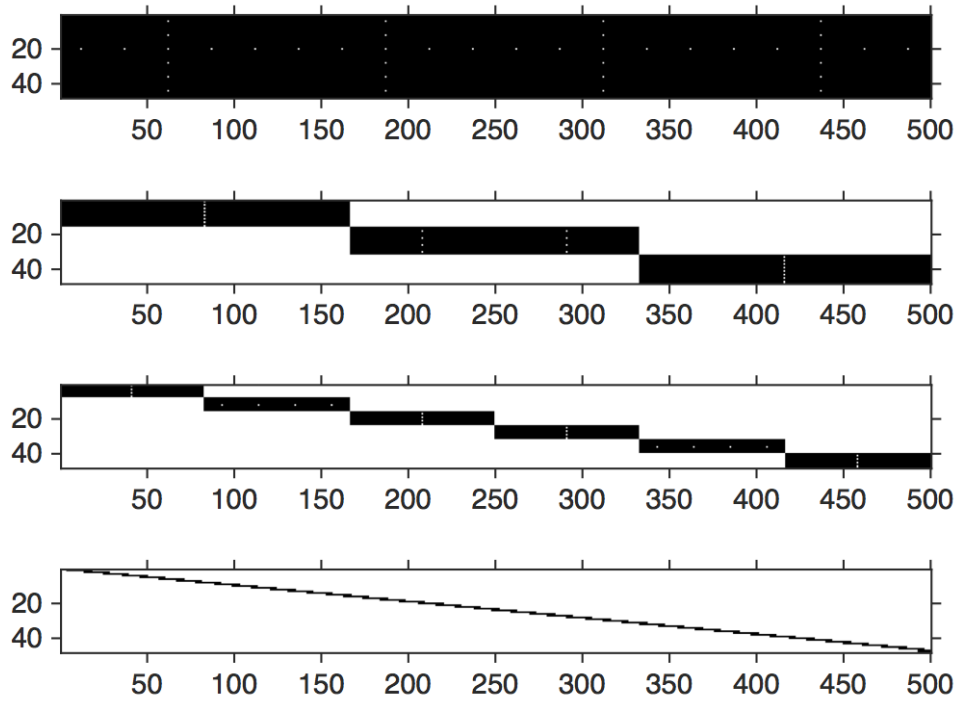


FIGURE 4.6. Structure of the Jacobian for one dimension of a single trajectory using different trajectory bases. Black denotes non-zero elements. From top to bottom: full DCT-type bases, DCT bases split into 3 parts, DCT-bases split into 6 parts, piecewise defined bases with small overlap. Total 48 bases each, 500 frames long trajectories.

4.2.3. Representational Power

How accurate a given trajectory t can be approximated by a linear combination of trajectory bases depends on the number of used bases K and the characteristics of the trajectory itself. To empirically evaluate the representative power of our formulation, trajectories were created. Figure 4.7 shows an example of such a generated trajectory together with the best possible approximation using trajectory bases given certain values of K.

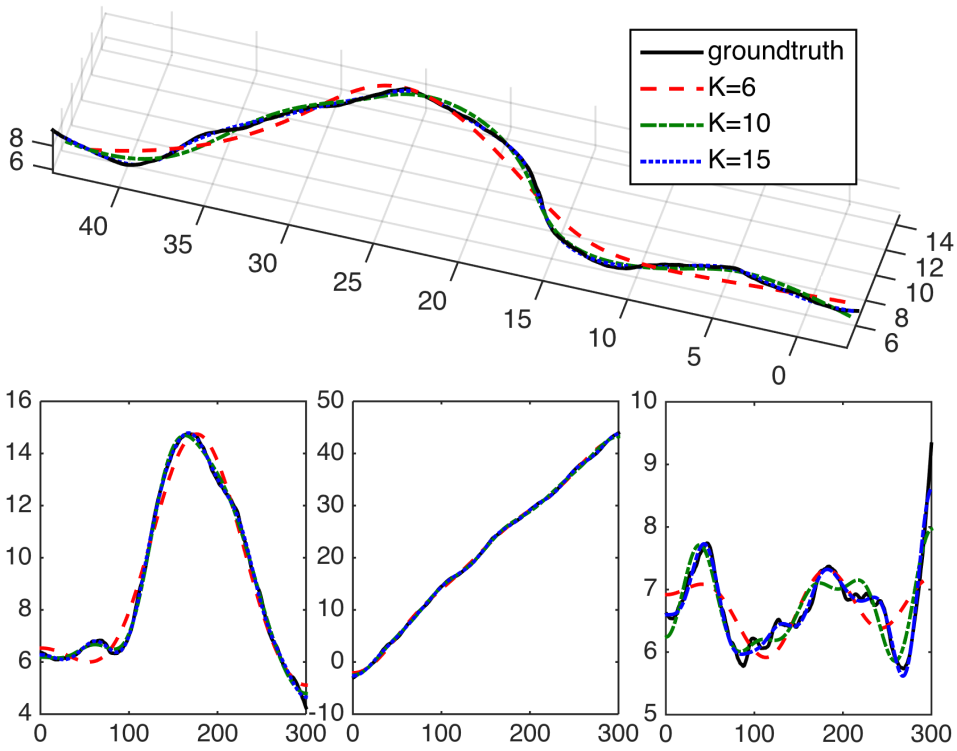


FIGURE 4.7. The black line is a trajectory, the other lines are the best possible approximation of the trajectory given different values of K.

The influence of K on the reconstruction accuracy was evaluated by creating a large number of different trajectories, reconstructing them by a set of K values and calculate the error for each case. Reconstruction accuracy $\mathbf{E}_{rec}$

is defined as the mean point to point distance between every point along every trajectory and its reconstruction ($\mathbf{E}_{rec} = \frac{1}{F \cdot P} \sum dist(\mathbf{t} - \mathbf{t}^{rec})$).

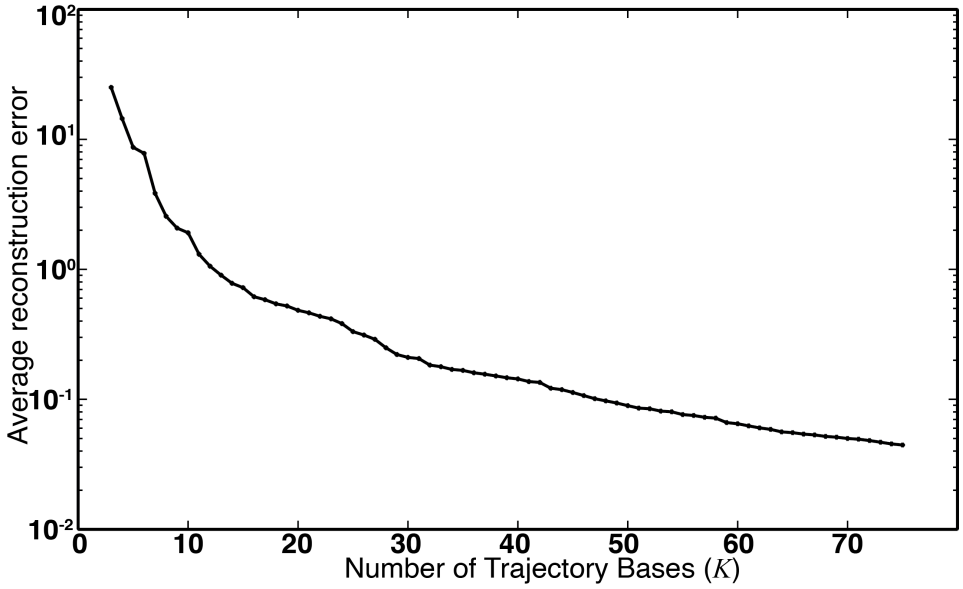


FIGURE 4.8. Mean reconstruction error (mean point-point distance) for different K values. Average over 1000 different trajectories, each approximated by a different number of trajectory bases ranging from $K = [3 \cdots 75]$.

With $dist(\cdot)$ being the Carthesian distance between each $3D$ point of the trajectory. Figure 4.8 shows the mean reconstruction error for different values of K. As expected, the higher the value of K becomes, the better the reconstruction gets.

However small values of K already yield good results. The actual best choice of K is difficult, there are tradeoffs between reconstruction accuracy and the introduction of possible ambiguities. Choosing a high K value reduces the smoothness assumption.

The question of picking the best K was discussed in detail in the NRSfM community [11, 77, 10, 5, 8, 4, 17], but without a clear conclusion on how to best estimate it. Almost all publications hand-pick the optimal value for each dataset evaluated, or give results on a set of different K values. The effect of K on the reconstruction is similar for BA and NRSfM since both aim to reconstruct a similar type of scenario. We pick K based on the length of trajectory at hand, but will not further discuss ways to choose K, since it is out of the focus of this work.

4.3. Incomplete Measurement Matrix

The previously discussed dynamic environment was assumed to be fully observed (full measurement matrix $\mathbf{Z}$) and therefore $\mathbf{S}_{dyn}$ is full, meaning that in every frame the position of every feature point is observed. In real world scenes most points will not be visible for the whole sequence. Features can leave the camera's field of view due to their own motion, the camera can move away from a feature, or features can become occluded by objects in the scene. We aim to only reconstruct the trajectory of a moving object for the time where observations exist. Unlike other methods such as NRSfM, the proposed dynamic scene bundle adjustment has no problems with these environments.

To represent trajectories of various lengths in an efficient fashion trajectory bases θ of variable length are required. Instead of using the same number of shape basis (K) for every trajectory, different values of K for each based on the length of the trajectoy $\mathbf{t}$ is used. This efficiently reduces the

parameter space to be estimated, since objects with a short trajectory are represented by a smaller number of trajectory coefficients.

$$\mathbf{t}^p = \begin{bmatrix} \theta_t^1 \cdot a_{xp}^1 + \cdots + \theta_p^{K_p} \cdot a_{xp}^K \\ \theta_t^1 \cdot a_{yp}^1 + \cdots + \theta_p^{K_p} \cdot a_{yp}^K \\ \theta_t^1 \cdot a_{zp}^1 + \cdots + \theta_p^{K_p} \cdot a_{zp}^K \end{bmatrix} \tag{4.3}$$

For trajectories with different lengths, we use the following notation: $\mathbf{t}^p$ where p denotes the index of the trajectory: p ranges from 1 to the number of trajectories $\mathbf{P}$. The length L of each trajectory $\mathbf{t}^p$ is $len(\mathbf{t}^p)$. Each trajectory is represented by three trajectory coefficient vectors a_{xp}, a_{yp} and a_{zp} – each K^p long. The coefficients relate to their corresponding trajectory basis θ^p. Stacking the trajectory coefficients into a matrix $\mathbf{A}^p$, and the bases into Θ, a trajectory can be represented in the form

$$\mathbf{t}^p = \Theta^p \mathbf{A}^p. \tag{4.4}$$

To represent $\mathbf{S}_{dyn}$ in the case of an incomplete measurement matrix a dynamic scene is the collection of all its trajectories.

$$\mathbf{S}'_{dyn} = \{\mathbf{t}^p\} \tag{4.5}$$

The value of K to represent each trajectory can be adjusted on its length and depending on additional knowledge about the moving object if available. The efficient representation discussed in this section is the key insight that allows reconstruction of real-world dynamic scenes using bundle adjustment.

4.4. Camera Representation

The camera position and orientation for each time step is part of the state vector. They can be directly represented by $3 \times F$ variables for an affine camera model, or $6 \times F$ for a perspective one. Since it is safe to assume that a camera is a physical object with mass, it is possible to reduce the positions to a trajectory in the same way as done for moving objects. For camera orientations this is also possible, depending on the chosen parametrization. Orientation can be parametrized by Euler angles, a 3D vector, a 3×3 rotation matrix or quaternions.

Here we use an angle-axis representation of the orientation. This allows efficient calculation of the rotation, and is sufficient since all scenarios evaluated are recorded with cameras that move on the ground. Therefore rotation is limited and special cases such as gimbal lock are unlikely to occur.

The number of bases K used to represent camera motion, specially its orientation can be higher than the ones used to represent moving objects. A higher number of bases is desireable because the effect of not being able to exactly approximate a camera trajectory will lead to more severe effects than for the observed object.

Given the same numbers used previously to illustrate the dimensionality of the problem at hand ($F = 150, P = 20, K_{traj} = 12$), we use a larger K of 20 to represent the camera pose. With that we can reduce the previous 6000 observations vs. $720 + 900$ unknowns to 6000 vs. $720 + 120$ for the affine case and $720 + 240$ for a perspective camera with known intrinsics (affine camera 2×3 camera matrix, perspective 3D pose + 3×3 for orientation).

4.4.1. Multiple Cameras

Using observations from multiple cameras can benefit the reconstruction. Generally depth reconstruction from a single camera is difficult and in some scenarios not possible. Having two or more cameras looking at the same scene at one time instance defines in theory the 3D structure of the scene in relation to the camera positions up to a scale factor. [33]. The previously discussed representation of a dynamic scene does not change in presence of multiple cameras. Important to note is that correspondence between the cameras is required, otherwise each camera defines its own independent scene. A correspondence is the observation of one feature in multiple cameras. For moving points the relative times at which the observation were made are required, for static points they are not relevant.

CHAPTER 5

Reconstructing Dynamic Scenes

AFTER introducing a compact efficient representation for dynamic scenes this chapter discusses how the representation is used to actually reconstruct the scene and how additional information about the scene can be included. Such priors might be the information which points are static or moving, how they are moving or if they belong to the same physical object. After that ways to find reasonable starting conditions for the bundle adjustment step are introduced. Next requirements for reconstructability are elaborated and different usage scenarios listed.

5.1. Error Function

First we define some terminology: $obs(\mathbf{t})$ is a vector of all camera observations of trajectory $\mathbf{t}$. $obs(\mathbf{t}, f, c)$ is a single observation (point on the image plane) at frame f seen by camera c.

We aim to minimize the reprojection error. For each observation made from each camera for each point along each trajectory the error is defined as

the distance between the projection of the reconstructed 3D point with the corresponding observation.

$$\min \sum_{c=1}^{\mathbf{C}} \left(\sum_{p=1}^{\mathbf{P}} \left(\sum_{f=1}^{\mathbf{F}_p} (proj(\mathbf{t}^p, \mathbf{C}_{pose}(c,f), \mathbf{C}_{int}(c)) - obs(p,f,c)) \right) \right)$$

using Equation 4.4 we get

$$\min \sum_{c=1}^{\mathbf{C}} \left(\sum_{p=1}^{\mathbf{P}} \left(\sum_{f=1}^{\mathbf{F}_p} (proj(\Theta^p \mathbf{A}^p, \mathbf{C}_{pose}(c,f), \mathbf{C}_{int}(c)) - obs(p,f,c)) \right) \right) \quad (5.1)$$

The parameters to be found are the trajectory coefficients $\mathbf{A}^p$, the camera poses $\mathbf{C}_{pose}(c,f)$ and the intrinsic camera parameters $\mathbf{C}_{int}(c)$.

5.2. Priors

This section introduces different priors that can help solve the Dynamic Scene Bundle Adjustment problem. Exploiting the structure of the problem, or utilizing other information can aid in restricting the search space for the optimization, leading to faster convergence and higher accuracy. These priors can be included by additional terms in the error function or can be included in the chosen representation.

$$\begin{aligned} \min \quad & \left(\sum_{c=1}^{\mathbf{C}} \left(\sum_{p=1}^{\mathbf{P}} \left(\sum_{f=1}^{\mathbf{F}_p} (proj(\Theta^p \mathbf{A}^p, \mathbf{C}_{pose}(c,f), \mathbf{C}_{int}(c)) - obs(p,f,c)) \right) \right) \right. \\ & \left. + prior \quad terms \right) \end{aligned}$$

5.2.1. Static Points

In general we do not assume knowledge of whether a feature is moving or static. In case it is known that a feature is static, the proposed representation is, without any modification, able to benefit from that information. By setting K to 1 for a feature it is defined as being static. To be able to do so, the first trajectory basis in the chosen trajectory basis representation has to be nonzero and linear, which is the case for the chosen DCTs.

Having the knowledge of static points in the scene strongly helps reconstructing the camera orientation and position more accurately, which in return also helps in reconstructing the trajectories of moving objects.

5.2.2. Planar Motion

Most things, like cars, bikes, ships or persons, moves on the ground – which is often *flat*. The fact that the motion is bound to the ground is a strong restriction to the movement of an object, efficiently giving it different motion patterns along the two dimensions spanning the ground plane and the one normal to that plane. Our approach represents each of the 3 dimensions of a trajectory by a separate linear combination of basis vectors.

The number of parameters used to reconstruct planar motion can be reduced by assuming that the dimension for height is changing less than other dimensions and can therefore be represented by a lower value of K or by $K = 1$ which assumes a fixed height above ground. Utilizing the fact of planar motion requires a coordinate system properly aligned with the ground

plane, meaning one axis is parallel to the normal of the ground plane. This can be achieved by properly aligning the initial camera trajectories.

5.2.3. Constant Distance

Moving objects are often rigid, or partially rigid objects such as cars, scooters or bicycles. Each of these moving objects can generate several feature tracks leading to several reconstructed trajectories for one moving object. In such cases the 3D distance of two trajectories belong to the same moving rigid object can be assumed constant. For the every point in time in which the two trajectories are reconstructed their 3D distance has to be the same. Even the distance is not known, it can be modeled with a single parameter in the Bundle Adjustment. Given multiple trajectories on one object several constant distance priors can be introduced. The error term for this prior can be written as follows

$$\min \sum_{f=1}^{F} \left(dist(\Theta_f^i \mathbf{A}_f^i, \Theta_f^j \mathbf{A}_f^j - d_{i,j} \right)^2. \tag{5.2}$$

Where F here is the range of frames for which both trajectories are defined.

Object detection or grouping of close, similar trajectories can create these correspondences. In practice the benefit due to this prior is huge. The reason for the benefit from this prior is twofold. First given observations of two points from one camera at one frame restricts the 3D location of the points only to lie along the beams thru camera center and the point on the image planes. Knowing the 3D distance between the 3D points significantly

reduces the possible 3D locations. Secondly in case of multiple trajectories where some are noisy or inaccurate at portions of their length the constant distance allows to correct for their errors. This is due to the fact that the majority of the other trajectories will be more correct.

5.2.4. Other Priors

Other ways to restrict the optimization space is to add bounds on change of point-point distance of camera or moving object trajectories. These priors represent knowledge about the speed and/or acceleration of moving objects allowing to determining these bounds. For road traffic scenes the maximum speed of cars can be used as prior, together with a maximum acceleration. These Priors require that a scale factor is determined first.

5.3. Initial Estimates

All discussed bundle adjustment approaches have in common that they require a starting point for their optimization, an *initial guess*. In the hypothetical ideal case in which no ambiguities are present, any starting point will suffice. But commonly ambiguities exist, requiring that the starting point is close enough to the solution, so the optimization will approach the true value. For three different sets of unknown parameters initial conditions are needed: the camera pose, camera intrinsics and the structure of the scene. Several methods exist that can be used to generate these initial estimates.

The intrinsic camera parameters can be found by calibration using a well defined physical object visible in one or multiple views. Several methods exist [81, 33] to perform this calibration. We have calibrated the cameras used, and refine the parameters during optimization to compensate for changes due to vibrations or temperature change.

Structure from Motion approaches can yield structure together with the camera poses. Given that the scene contains enough static object this will lead to good results. Multi-body SfM is an excellent candidate for creating starting points in scenes with several large objects (objects that have a large number of features on them). NRSfM, especially the trajectory based version, can create starting conditions in scenarios with outlier free low noise data.

Visual Odometry [6, 19] is a method to estimate a cameras trajectory based on an image sequence. State of the art implementations [43] are fast, reasonably accurate and are able to handle dynamic environments. Given the estimated camera poses, trajectories can be initialized using triangulation. Then trajectory basis parameters are estimated using Discrete Cosine Transformation. Given there is enough camera motion, and that the camera motion is not correlated with the motion of the object, this method of generating a starting position for the BA procedure is sufficient. The triangulation will not be highly accurate, especially not for moving objects since the objects will change their position between the images. In the presented scenarios starting points were generated this way.

5.4. Reconstructability

Bundle adjustment for dynamic scenes suffers from the same reconstructability problems like all image based reconstruction approaches. When the camera is not moving sufficiently, it is difficult to make reliable reconstructions. Also when the motion of dynamic objects is correlated to the cameras movement, or an object is moving along the optical center of the camera, no depth estimation is possible.

Since we deal with dynamic scenes the camera motion is of great importance. Park [55] defined a measure of reconstructability for NRSfM, that, despite the different approaches of NRSfM and BA, still holds true to some extent. It states that for a successful reconstruction the motion of the camera has to be large compared to the objects motion (or deformation). To which extent this will cause issues is one point that will be explored more. Knowing these facts we want to state clearly that we are aware that reconstruction of a dynamic scenes from a single moving camera is not possible in all scenarios.

CHAPTER 6

Experimental Results

THIS chapter presents experimental results. First an overview over different scenarios evaluated and an discussion of the applicable priors is given. Then simulated data is used to elaborate the effects of different scenario parameters such as number of moving objects, number of static objects, scene length, and different starting conditions. The simulated data allows exact control of the scenario, of noise levels and the motion of objects and comparison to NRSfM methods. After that several reconstructed real world scenarios will be presented and analyzed.

6.1. Categories

Three different scenario categories are established and differentiated. Each requiring different conditions and priors to make them solvable.

6.1.1. Single Fast Moving Camera

Reconstruction of moving objects from a single camera generally requires that the motion of the camera and that of the object are not correlated, and that the camera is moving fast compared to the object. These types of scenario are often evaluated in NRSfM research. The image data is regularly synthesized from simulation.

We have successfully solved these types of scenario from simulated data using affine and perspective camera models. Finding real world scenarios that are not unrealistic 'toy-problems' and satisfy the requirements are hard to find and get.

6.1.2. Overlapping Field of View

In this scenario it is assumed to have two or more cameras present that move independently, but have a overlapping field of view most time. Real world examples would be cameras mounted on a scooter drivers helmet and the scooter, or cameras in two cars driving beside each other for some time. Also any form of stereo camera, specially when the calibration is not exactly known or wrong falls in this category.

The overlapping field of view allows to establish feature correspondences for static as well as moving objects. Any additional prior helps to improve reconstruction. This type of scenarios has been successfully reconstructed and will be presented in the next section.

6.1.3. Independently Moving Cameras

Similar to the previous scenario, but the field of view is only expected to overlap at short time intervals. Only feature correspondences for a few static points are available that allow alignment of the camera trajectories, but no correspondences are available for moving objects. This represents any typical traffic scenario recorded by several cameras. To solve it in real world conditions prior knowledge about static points and constant trajectory distance priors are required. The last presented real-data scenario falls in this category.

6.2. Simulated Data

A dynamic 3D scene is created with moving objects following randomly generated realistic trajectories. Image data is derived from this 3D scene using affine and projective camera models. Image noise is added to features, as well as outliers (wrong feature correspondences). The advantages of using simulated data initially is that exact ground truth data is available to estimate the performance and that the camera/object motion can be exactly controlled. This way completely observed scenes can be used which are difficult to create in real scenarios and are necessary for comparison with NRSfM methods.

6.2.1. Simulated Scenarios

A series of experiments has been performed to verify the proposed approach. The synthetic dynamic environment is created from a number of moving objects with random starting positions, moving according to a motion model that is steered by random inputs and a number of randomly placed static points. The motion model used creates realistic motion patterns – meaning that angular and directional accelerations are in the range of what real world physics permit.

The images below show two sample scenarios. The first (Figure 6.1) shows two typical scenarios used for evaluation. The two differ in the number of objects and the temporal length of the scene. The second (Figure 6.2) is a similar scenario, but objects are restricted to move in two dimensions only. This is an approximation of a typical traffic scene in which vehicles drive on flat ground.

6.2.2. Evaluation

The reconstruction accuracy is evaluated after the the models are aligned. To align the models Procrustes analysis [27, 22], a form of statistical shape analysis, is employed. For a meaningful comparison of the reconstructed scene to its ground truth the two need to be optimally superimposed first. They need to be translated, rotated and scaled. This is due to the fact that reconstruction is only accurate up to a scale factor, and the reconstruction result can lie in a different coordinate system.

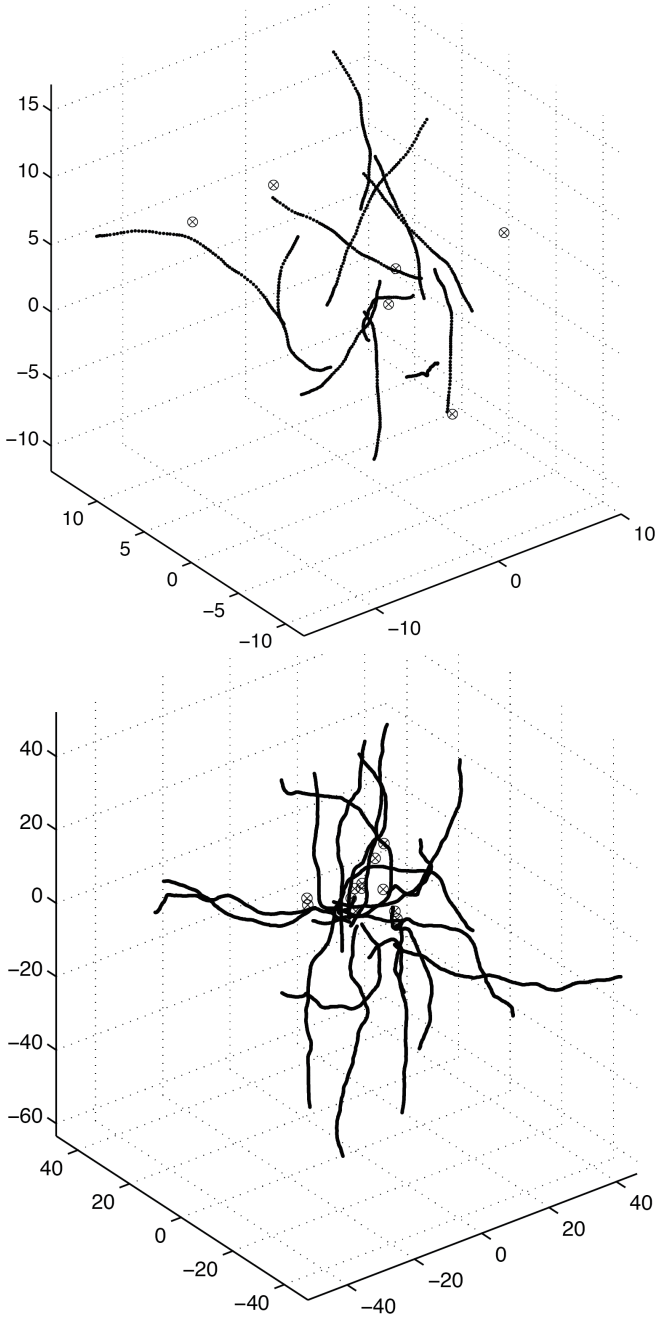


FIGURE 6.1. Sample scenario of a dynamic scene. Lines represents trajectories of moving objects, static objects are depicted with a circled x. On top a small scenario consisting of 12 moving objects and 6 static ones, 250 frames long. Below a scene consisting of 18 moving objects and 12 static points, 500 frames long.

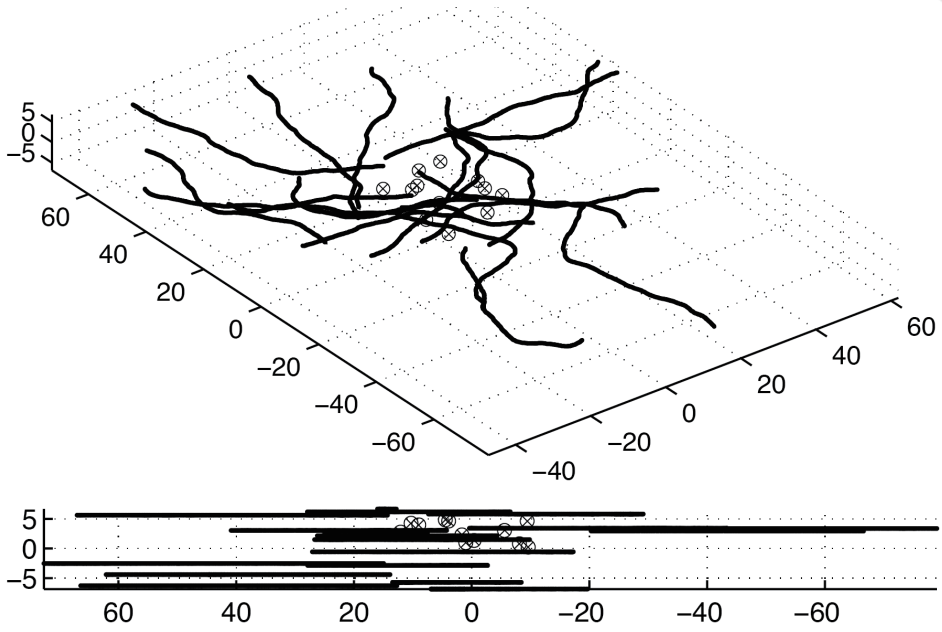


FIGURE 6.2. Planar motion scenario. Lines represents trajectories of moving objects, static points are visualized with a circled x. The scene consists of 18 moving objects and 12 static ones and spans 500 frames. Below the scene can be seen from the side. In this scenario motion is restricted to two dimensions.

The two shapes are normalized so that the root mean square distance (RMSD) from points to the center is 1. The shape distance, which in this case, we call reconstruction error $\mathbf{E}$, is the sum of squared distances (SSD) between corresponding points. Figures 6.3 and 6.4 show several reconstructions with their respective reconstruction error. They serve as a reference point on the range of reconstruction error.

6.2.3. Reconstruction Results

The proposed approach has been evaluated using the scenarios just described. Different values for K have been used to represent trajectories

and camera motions. Current evaluations use an affine projection camera model, resulting in 3 parameters to estimate for each camera pose. No prior information about the motion state of any object is used, meaning all objects are treated moving, even though not all objects move. No priors whatsoever are used, except the assumptions inherent in the trajectory based representation. The scene is fully observed and optimization is performed by solving the nonlinear least squares problem using the Levenberg-Marquardt method.

Below in Figures 6.3 and 6.4 several reconstruction results are shown. Scene lengths vary, as well as number of objects and the initial conditions used to create them. Later we will detail the effect of several of the parameters.

6.3. Analysis and Comparison

The proposed method is compared to the NRSfM method of Akhter et al[4] and evaluated with various parameters. To do so 1000 scenes of different size have been created and reconstructed using the proposed Dynamic Scene Bundle Adjustment with different starting conditions and two NRSfM methods.

6.3.1. Initial Estimates

Four different types of initial conditions have been evaluated. First the ground truth was used with added noise, labeled GT+noise1 and GT+noise2. The first referring to the use of ground truth trajectory coefficients with small noise $\mathbf{A} \cdot \mathcal{N}(\mu = 1, \sigma^2 = 0.1)$, and the second to the same with more

noise ($\mathcal{N}(\mu = 1, \sigma^2 = 0.25)$). Lines refers to the use of a line as approximation of the trajectory as initial guess. The motivation for this is, that given an initial reconstruction of the camera path using visual odometry or other methods as discussed in section 5.3, a few frames can be used to get a rough triangulation of points along the trajectory. Between the points a linear interpolation can be used leading to lines. Lines + noise1/2 stands for different noise added to the two points defining the line. Figures 6.5 and 6.6 show examples for the initial estimates introduced. Random stands for initializing $\mathbf{A}$ as $\mathcal{N}(0, 1)$.

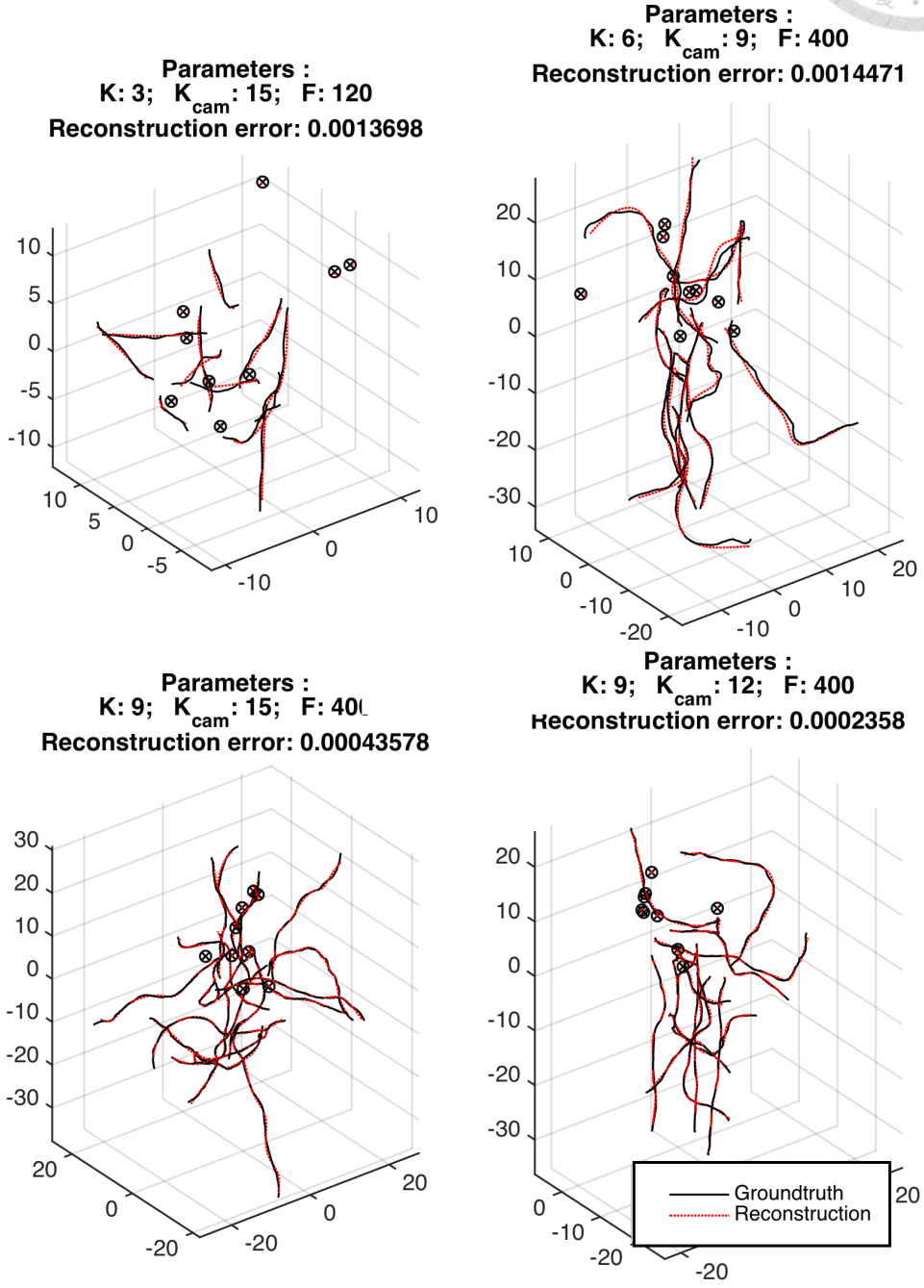


FIGURE 6.3. Successful Reconstructions for different simulated scenarios and parameters. Black lines represent the ground truth trajectories of moving objects, black circles static points. The dotted (red) lines are the reconstruction results.

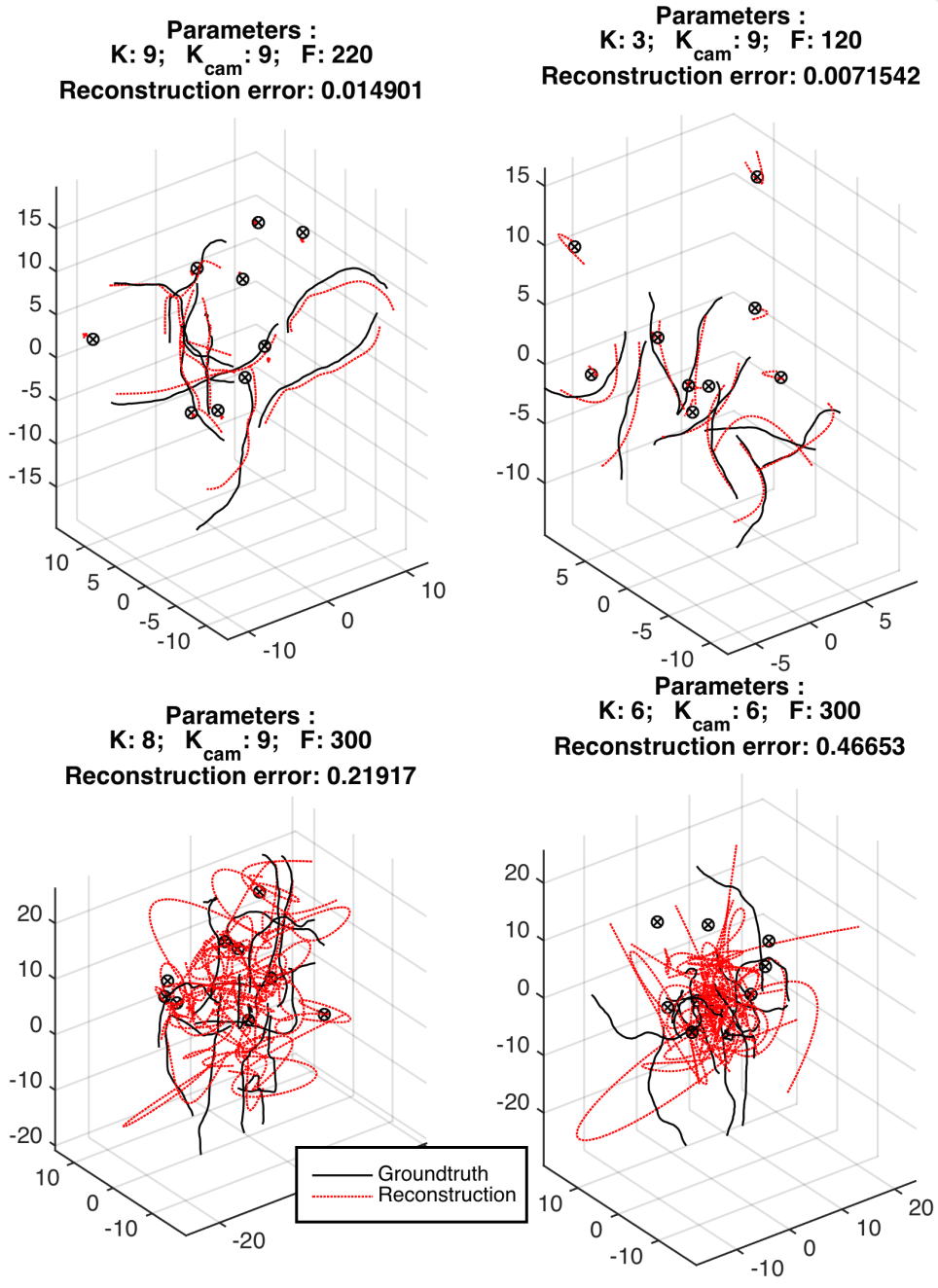


FIGURE 6.4. More reconstruction results. Black lines represent the ground truth trajectories of moving objects, black circles static points. The dotted (red) lines are the reconstruction results. Bottom row are failure cases. Failure due to bad, randomly generated starting conditions which the method can not handle.

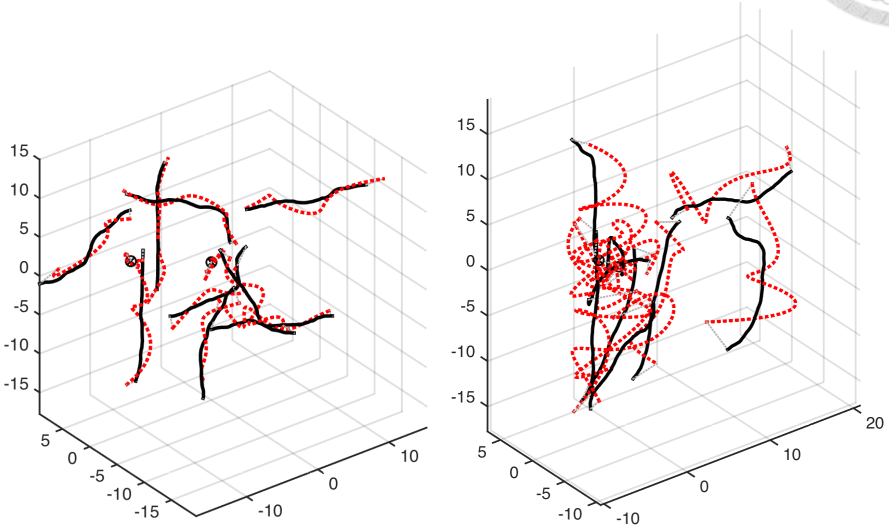


FIGURE 6.5. Examples of initial estimates used. Black lines represent ground truth, black circles static points. Dotted lines are the created initial guesses. Left side represents GT+noise1 ($\mathbf{A} \cdot \mathcal{N}(\mu = 1, \sigma^2 = 0.1)$), right side GT+noise2 ($\mathbf{A} \cdot \mathcal{N}(\mu = 1, \sigma^2 = 0.25)$).

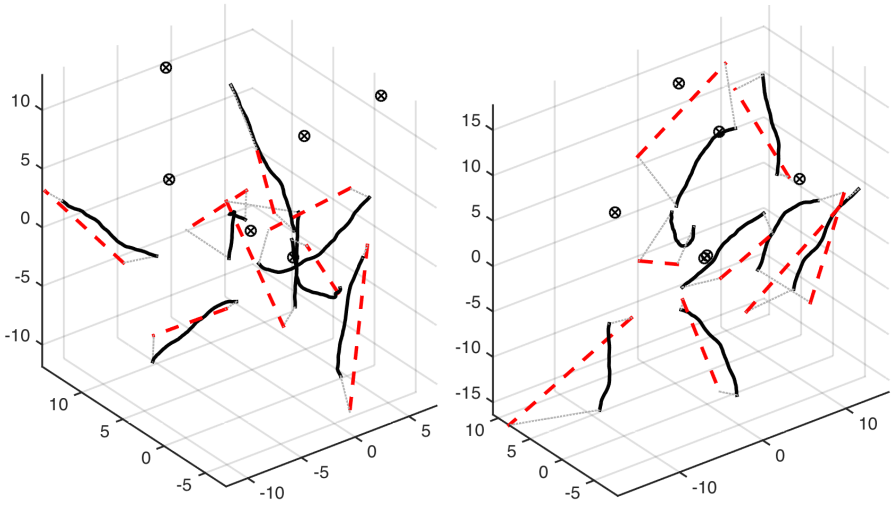


FIGURE 6.6. Examples of initial estimates used. Black lines represent ground truth, black circles static points. Dotted lines represent the corresponding initial guess. Left side ground truth with small added noise to the lines. Right side more noise added.

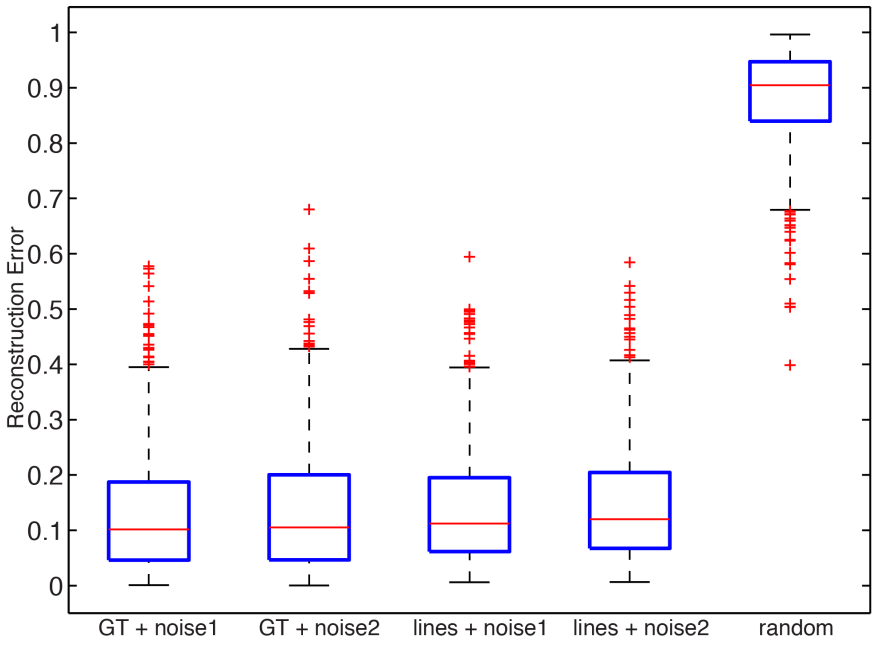


FIGURE 6.7. Box plot for different methods and initial conditions. Based on 1000 reconstructions of generated scenarios. The central mark in the box is the median, the edges of the box are the 25th and 75th percentiles. Whiskers extend to the most extreme data points not considered outliers (+/- 2.7σ). Outliers are plotted individually as crosses. GT+noise1/2 refers to ground truth with noise added as referred to in the text, same for lines. Random indicates that $\mathbf{A}$ is initialized with $\mathcal{N}(0, 1)$.

6.3.2. Comparison to NRSfM

The proposed method is compared to the NRSfM approach of Akhter et al. [4]. Another approach evaluated was using the NRSfM result as input to the proposed Dynamic Scene Bundle Adjustment framework.

The comparison is based on 1000 trials. For each trial a scene with a varying number of static and moving points and varying number of frames was created. A camera path was created and from it affine images were calculated. Image noise was added, and based on these images the scene was reconstructed using the proposed method given the previously discussed starting conditions, as well as the NRSfM based methods. The reconstructed scene was compared to the ground truth after aligning with procrustes analysis. Figure 6.8 shows the results for the different methods and initial conditions in a boxplot.

NRSfM is often not able to reconstruct the scene, and using its result as input for the proposed method does not significantly improve results. An analysis of the results gave the insight that in case the NRSfM fails, using the wrong reconstruction as starting point for our method does not recover the failed reconstruction, but a good reconstruction can be further improved by our method, resulting in a better result for the combined methods.

Using a starting point for the Bundle Adjustment that is reasonably close to the solution yields much better results than NRSfM, and as expected the better the initial guess is, the better the final result gets. A completely randomly picked starting point is not able to solve the problem, yielding results worse then NRSfM.

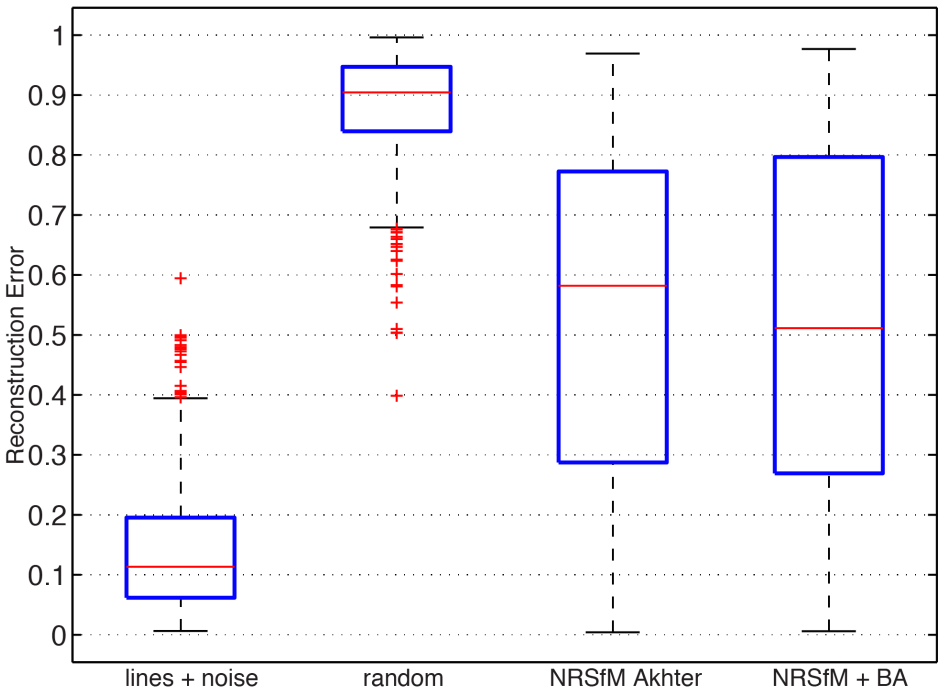


FIGURE 6.8. Box plot for different methods and initial conditions. Based on 1000 reconstructions of generated scenarios. The central mark in the box is the median, the edges of the box are the 25th and 75th percentiles. Whiskers extend to the most extreme data points not considered outliers (+/- 2.7σ). Outliers are plotted individually as crosses. Lines + noise refers to using a line approximation of the trajectories as initial guess, random fills $\mathbf{A}$ with random values. nrsfm Akhter et al. refers to [4], NRSFM+BA takes the result of the nrsfm factorization of Akhter et al. as initial guess to the Dynamic Scene Bundle Adjustment.

6.3.3. Effect of Multiple Cameras

The effect of multiple cameras was evaluated by generating three camera paths, synthesizing images from them, and then reconstructing the scene either using the images from one, two or three cameras. Figure 6.9 shows the error for the evaluated cases. As expected does the error decrease with added cameras.

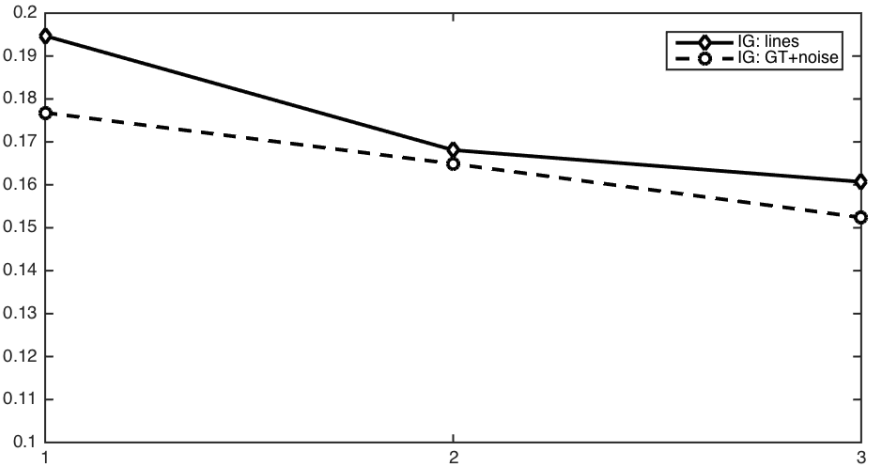


FIGURE 6.9. Effect of multiple cameras. Vertical axis is the reconstruction error. ig-GT refers to the use of ground truth with added noise as initial guess, ig-lines to the use of linear trajectory approximations with added noise as starting point.

6.3.4. Effect of Loss Functions

Loss functions were introduced in Section 3.4.3. Their goal is to robustify the optimization in the presence of image noise and outliers. Therefore loss functions were evaluated given different noise levels. Figure 6.10 shows the mean error for different noise levels.

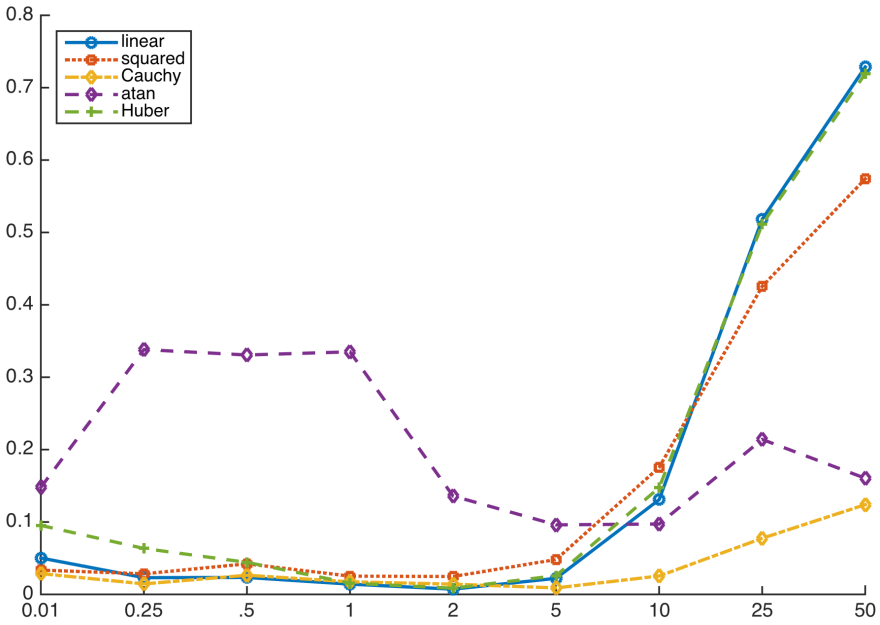


FIGURE 6.10. Effect of loss functions under different noise levels. Vertical axis represents the mean reconstruction error, on the horizontal axis increasing noise levels (noise level in pixels, normal distributed).

For the scenarios evaluated a Cauchy based loss function gives the best results. Huber and linear loss function behave almost similar, which was expected, given that for larger errors Huber is using the linear loss function. atan based loss functions only give an advantage for large error levels.

6.4. Real Data

All results shown before were based on simulated scenarios. The following real world sequences are used to verify the effectiveness of the proposed approach. Two different sets of cameras were used in different settings. First a sequence from a stereo camera rig is presented, followed by scenarios recorded from several wide angle action-cameras.

6.4.1. KITTI Stereo Sequence

The first real data sequence evaluated is from the KITTI vision benchmark suite [26]. This dataset was captured from a car with a roof mounted wide-baseline stereo camera. The sequence picked is short (158 frames), and includes two moving objects: a van and a person on a bicycle. Figure 6.11 shows the platform used to capture the data. The scenario fits to the "Overlapping Field of View" category described before.

In Figure 6.12 several frames from the sequence are shown. Detected and tracked features are displayed in the images. Coloring depends on their status - static points are colored blue, moving ones red and orange where red refers to single camera detection, and orange to detection in both cameras.

FIGURE 6.11. Autonomous platform AnnieWay that was used to capture KITTI dataset. Image from [26]

FIGURE 6.12. Frames 1, 52, 104 and 158 from the used KITTI image sequence. Blue points are static features, orange and red features that are moving. Orange points are features that are visible in both cameras.

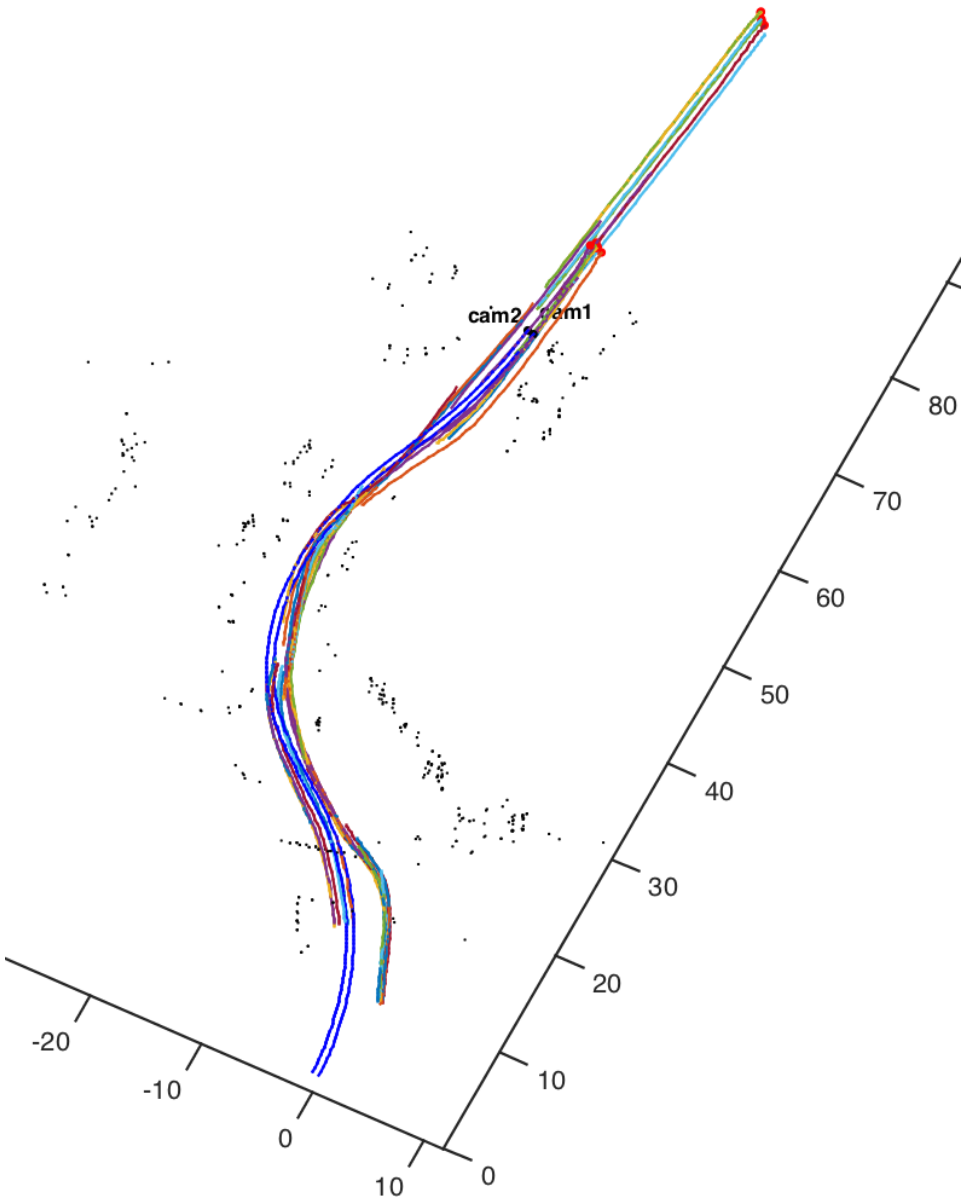


FIGURE 6.13. Reconstructed dynamic scene. Black points are static, the blue lines represent the trajectories of the two cameras. Differently colored lines ending with red dots are trajectories of moving objects. A clear grouping of the trajectories representing the two moving objects can be seen.

Figure 6.13 shows the reconstructed dynamic scene. The two blue lines represent the position of the stereo cameras over time. Their paths were reconstructed independently. Small black dots are static features, and colored lines represent reconstructed moving objects. The red dots are the 3D feature positions in the last frame. To show the reconstruction accuracy we look at the 3D distance between the two cameras (Figure 6.14). The distance is nearly constant. From this distance and the known true baseline a scale factor can be computed to allow metric reconstruction.

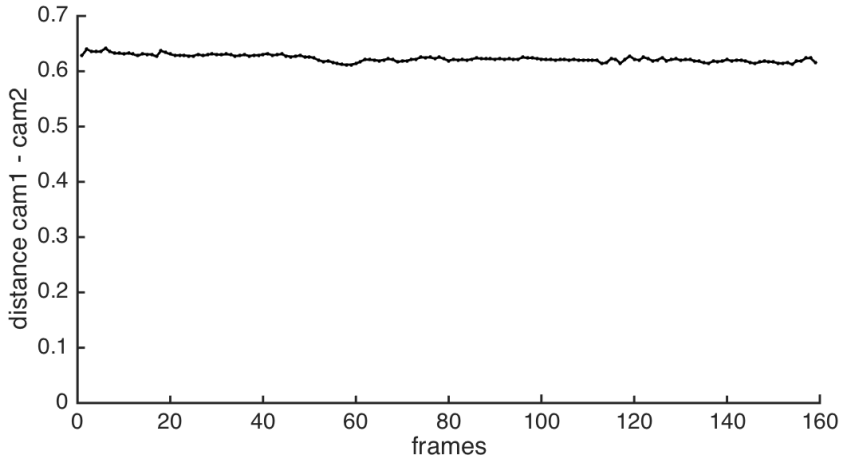


FIGURE 6.14. 3D distance between the two cameras over the sequence.

In Figure 6.15 details of the reconstruction at timestep $f = 20$ is shown. The reconstruction optimizes the whole scene as once, but in the figures the trajectories until frame 20 for better clarity.

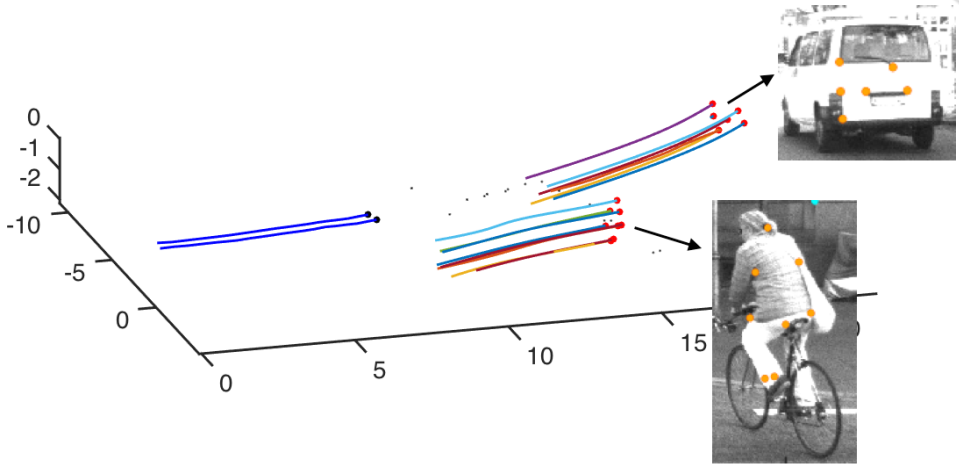


FIGURE 6.15. Reconstruction details seen at f=20. Observations made by camera 1 at the corresponding frame are overlayed.

6.4.2. Campus Sequence 1

The next scenario evaluated was captured on NTU campus from a bike. Two indenpendently moving cameras were used. One camera was mounted on the head of the person riding the bike, the other one on the bikes handlebars. The cameras used were wide-angle action-cameras (GoPro Hero 3). Figures 6.16 and 6.17 show frames from both cameras with detected features.

FIGURE 6.16. Frames from the used image sequence (Part 1). Blue points are static features, orange and red features that are moving. Orange points are features that are visible in both cameras.

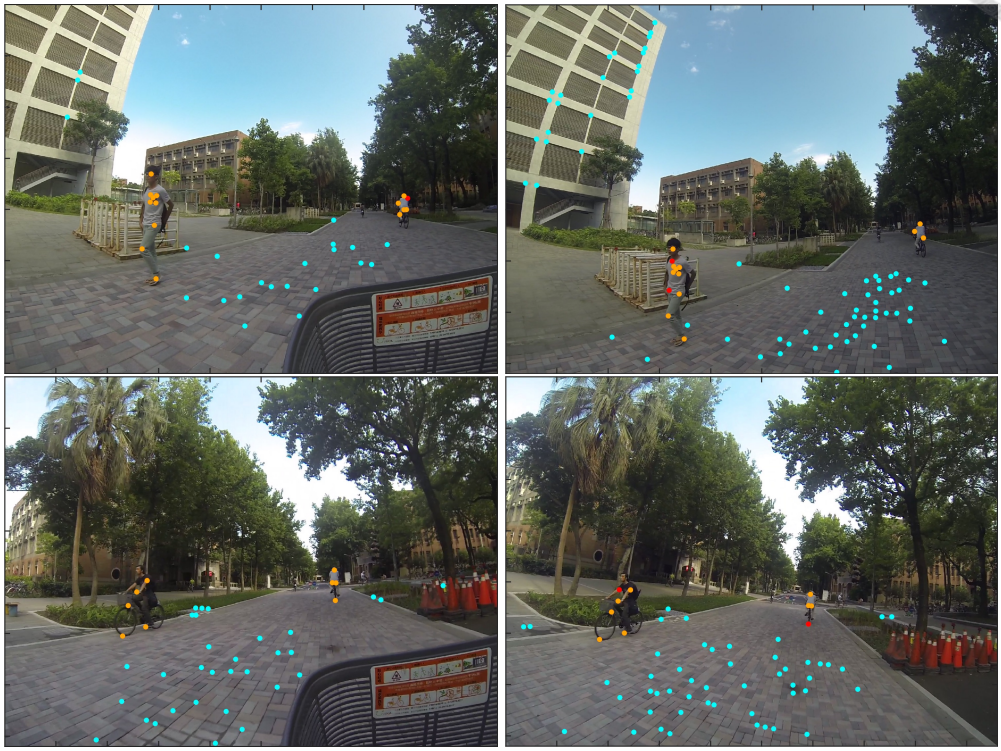


FIGURE 6.17. Frames from the used image sequence (Part 2). Blue points are static features, orange and red features that are moving. Orange points are features that are visible in both cameras.

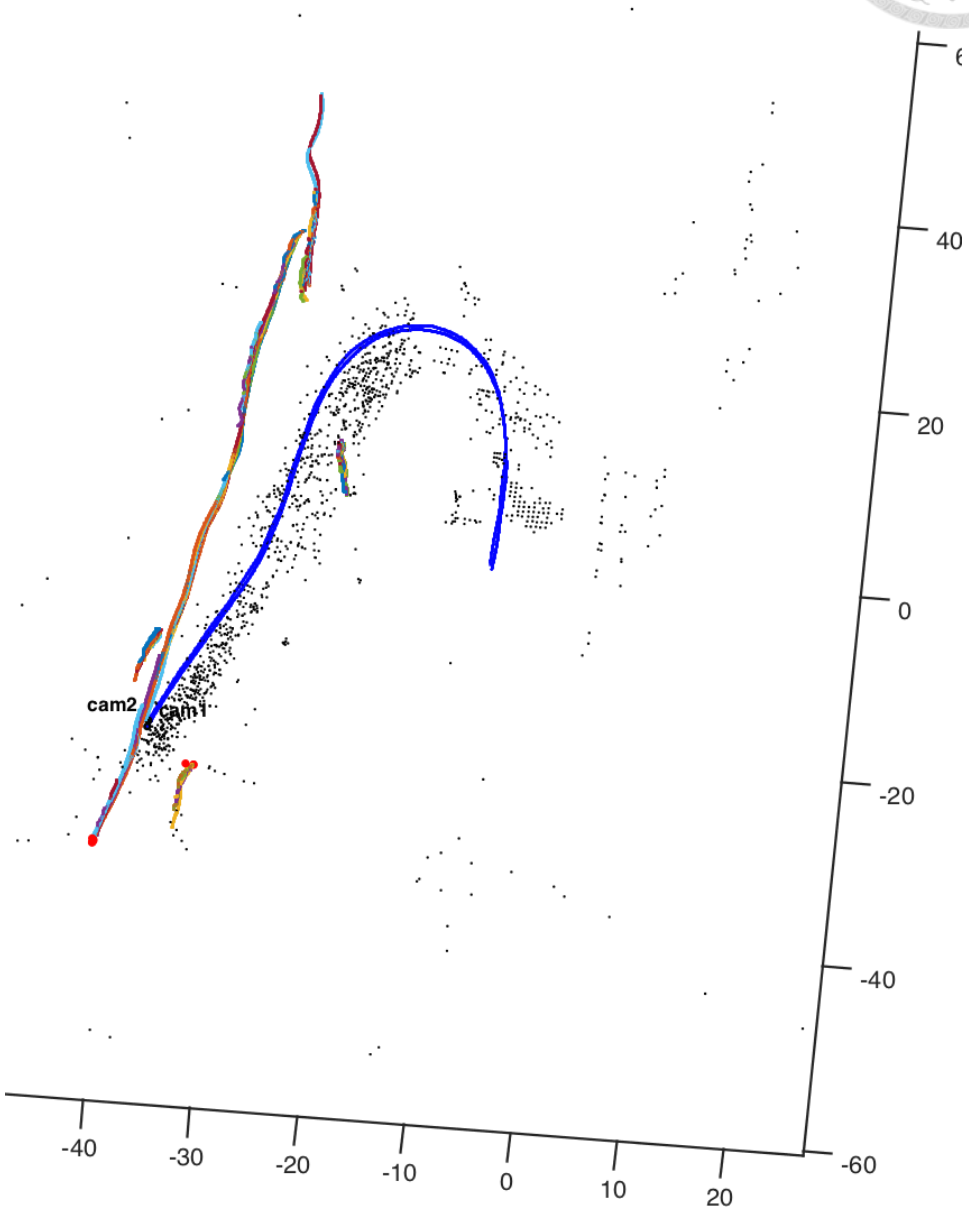


FIGURE 6.18. Reconstructed dynamic scene. Black points are static, the blue lines represent the trajectories of the two cameras. Differently colored lines ending with red dots are trajectories of moving objects. A clear grouping of the trajectories representing the two moving objects can be seen.

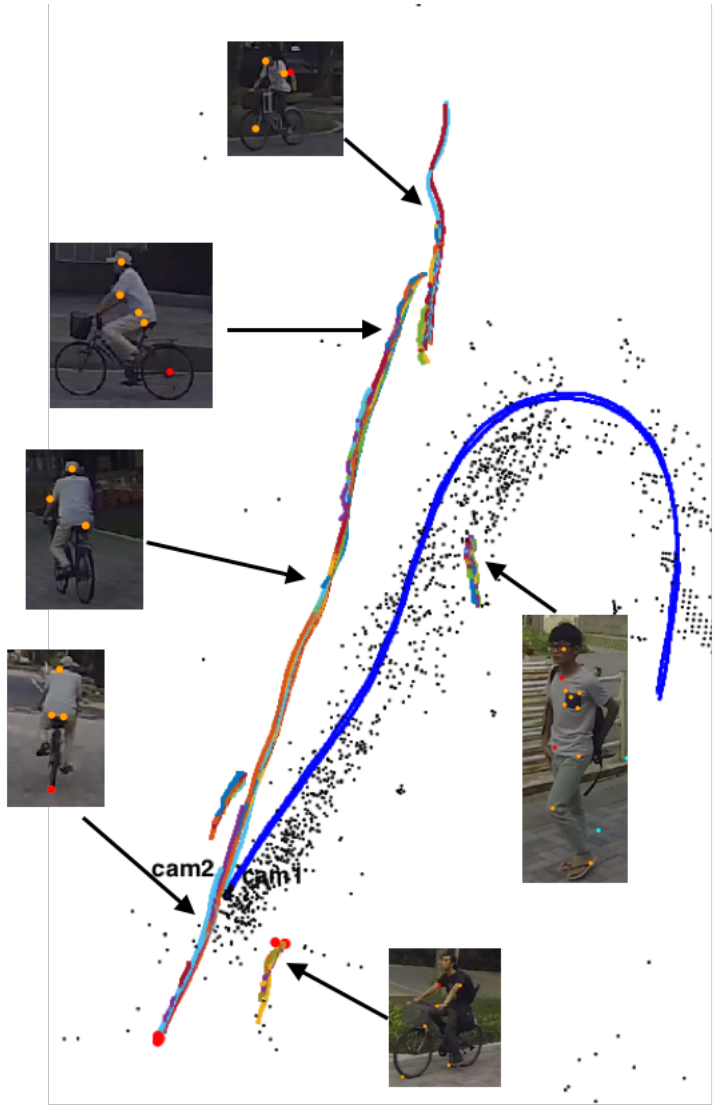


FIGURE 6.19. Details and annotations for the reconstructed dynamic scene.

6.4.3. Campus Sequence 2

This sequence is captured at the same area, but with 3 independent cameras which only have an overlapping field of view at some points in time, similar to the third mentioned category. Two persons on a bicycle with head-mounted cameras and one person walking captured the sequence. Figure 6.20 shows parts of the video with features marked.

For solving knowledge of static points was assumed, as well as the constant trajectory distance prior - meaning that it is known that trajectories belong to the same moving object. Some static feature correspondences are found, but no correspondences between cameras for moving objects. Each trajectory of a moving object is only visible from one camera. Figure 6.21 gives an overview over the reconstructed scene, Figure 6.22 shows details.

FIGURE 6.20. Left row are images from the 3 cameras taken at f=1, right at f=225. Blue points are static features, red features that are moving.

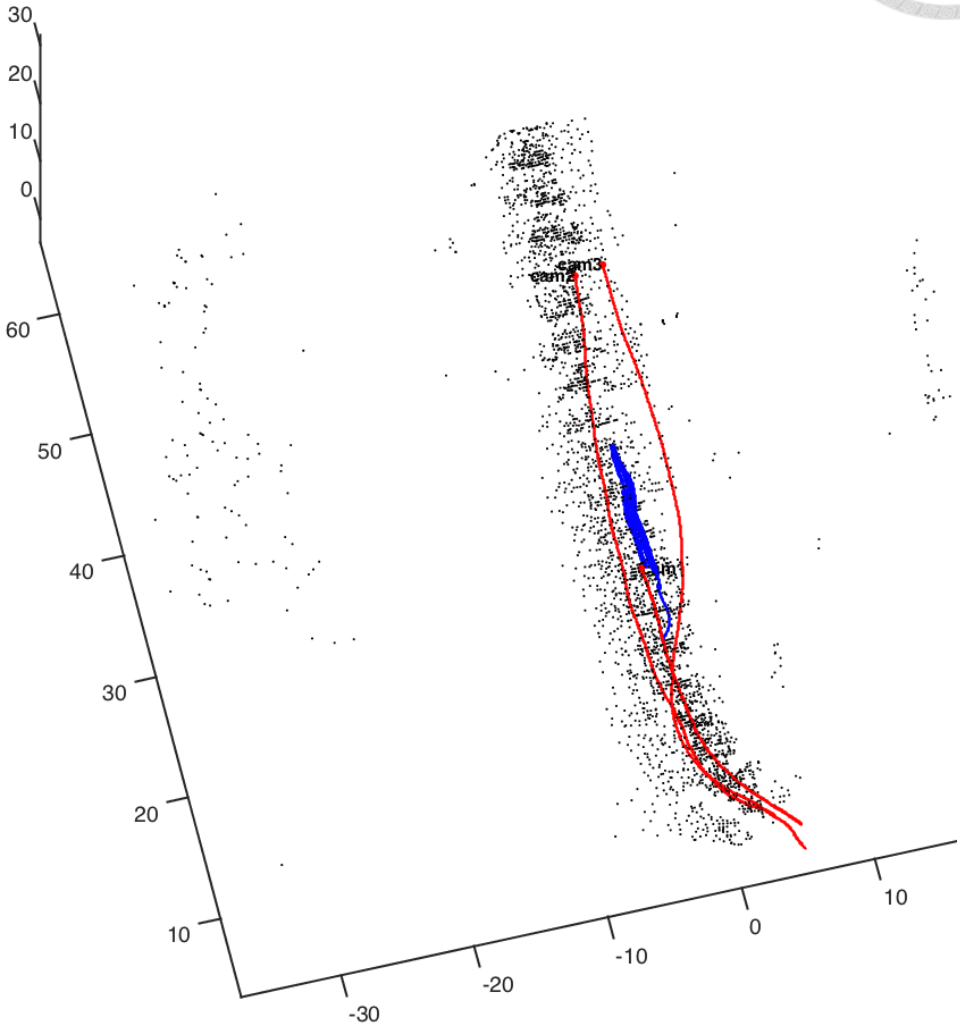


FIGURE 6.21. Reconstructed dynamic scene. Black points are static, the blue lines represents the trajectories on one moving object. The trajectories of the 3 cameras are red.

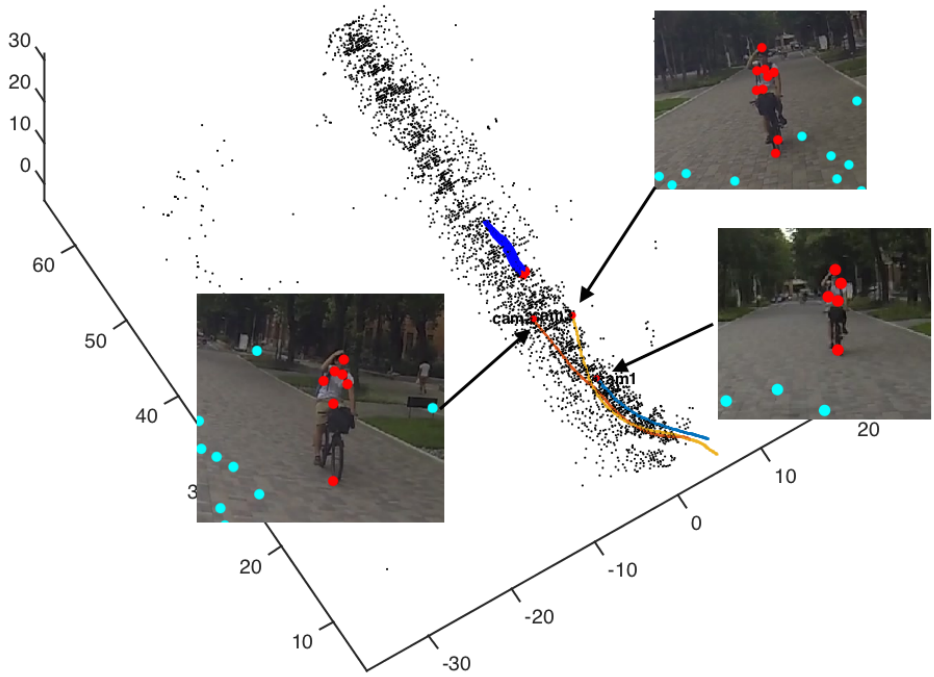


FIGURE 6.22. Detail of the reconstructed dynamic scene at frame 225. Overlayed parts of the images observed by the 3 cameras at this point. Red points mark detected features of moving points.

CHAPTER 7

Conclusion

We presented the extension of Bundle Adjustment to dynamic scenes. The image sequences of one or multiple cameras moving through a dynamic environment are used to reconstruct the path and orientation of the camera, the 3D position of static features as well as the 3D trajectories of moving ones. To make this possible, an efficient, low-dimensional representation of scene and camera motion was introduced. Based on a linear combination of predefined trajectory bases a compact formulation for fully observed scenes has been derived, and extended to scenarios with incomplete observations. Also compact representations for the camera poses have been introduced. Our method is, in difference to other approaches, able to deal with incomplete and noisy data. We require no knowledge about the objects, not even which are moving or static, although the inclusion of priors can improve reconstruction results. Static point priors can be seamlessly included in the given representation, other priors have been introduced and discussed and allow the reconstruction of challenging real world scenes.

The proposed approach has been experimentally verified based on an affine camera model with simulated complete measurement data to allow comparison to other methods. The experiments emphasize the power of our representation and outperform other methods. More experiments using real data have been performed using different perspective cameras in various highly dynamic scenes. The successful reconstruction of these dynamic scenes shows the effectiveness of our method.

BIBLIOGRAPHY

[1] Sameer Agarwal, Noah Snavely, Steven M. Seitz, and Richard Szeliski. Bundle adjustment in the large. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 29–42. Springer Verlag, Berlin, Heidelberg, 2010.

[2] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Kyoto, Japan, September 2009.

[3] Ijaz Akhter, Yaser Sheikh, Sohaib Khan, and Takeo Kanade. Trajectory space: A dual representation for nonrigid structure from motion. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 33(7), pages 1442–1456, July 2011.

[4] Ijaz Akhter, Yaser Ajmal Sheikh, Sohaib Khan, and Takeo Kanade. Nonrigid structure from motion in trajectory space. In *Conference on Neural Information Processing Systems (NIPS)*, December 2008.

[5] Del Bue Alessio. A factorization approach to structure from motion with shape priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2008.

[6] AnatLevin and Richard Szeliski. Visual odometry and map correlation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 611–618, 2004.

[7] K.B. Atkinson and J.B. Fryer. *Close Range Photogrammetry and Machine Vision*. Whittles Publishing, 1996.

[8] Adrien Bartoli, Vincent Gay-Bellile, Umberto Castellani, Julien Peyras, Søren Olsen, and Patrick Sayd. Coarse-to-fine low-rank structure-from-motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2008.

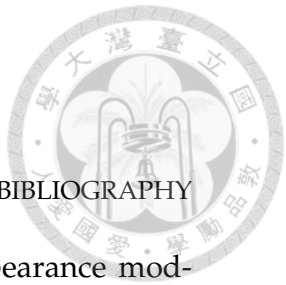
[9] Ake Björck. *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics, 1996.

[10] Matthew Brand. A direct method for 3d factorization of nonrigid motion observed in 2d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 122–128, June 2005.

[11] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 690–696, 2000.

[12] SuShing Chen. Structure from motion without the rigidity assumption. *IEEE 3rd Workshop on Computer Vision: Representation and Control*, pages 105–112, 1985.

[13] Javier Civera, Andrew J. Davison, and J. M. M. Montiel. Inverse depth parametrization for monocular slam. *IEEE Transactions on Robotics*, 24(5):932–945, October 2008.



[14] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 484–498. Springer Verlag, Berlin, Heidelberg, 1998.

[15] João Paulo Costeira and Takeo Kanade. A multi-body factorization method for motion analysis. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1071–1076, June 1995.

[16] Geoffrey Cross, Andrew W. Fitzgibbon, and Andrew Zisserman. Parallax geometry of smooth surfaces in multiple views. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 323–329, 1999.

[17] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2018–2025, June 2012.

[18] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005.

[19] Andrew J. Davison, Ian D. Reid, Nicolas D. Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 29(6), pages 1052–1067, 2007.

[20] Andreas Dopfer and Chieh-Chih Wang. What can we learn from accident videos? In *International Automatic Control Conference (CACS)*, pages 68–73, December 2013.

[21] Andreas Dopfer, Hao-Hsueh Wang, and Chieh-Chih Wang. 3d active appearance model alignment using intensity and range data. In *Robotics and Autonomous Systems*, volume 62, pages 168–176, February 2014.

[22] Ian L. Dryden and Kanti V. Mardia. *Statistical shape analysis*, volume 4. John Wiley & Sons New York, 1998.

[23] Olivier Faugeras and Quang-Tuan Luong. *The Geometry of Multiple Images: The Laws That Govern The Formation of Images of A Scene and Some of Their Applications*. MIT Press, Cambridge, MA, USA, 2001.

[24] Andrew W. Fitzgibbon and Andrew Zisserman. Multibody structure and motion: 3-d reconstruction of independently moving objects. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 1842, pages 891–906. Springer Verlg, Berlin, Heidelberg, 2000.

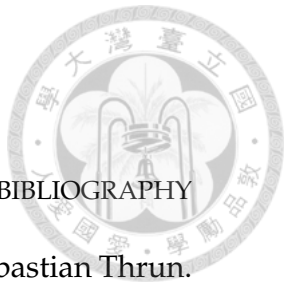
[25] Andreas Geiger. *Probabilistic Models for 3D Urban Scene Understanding from Movable Platforms*. PhD thesis, KIT, 2013.

[26] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[27] John C. Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.

[28] S. I. Granshaw. Bundle adjustment methods in engineering photogrammetry. *The Photogrammetric Record*, 10(56):181–207, 1980.

[29] The Guardian. Apple maps: how google lost when everyone thought it had won. `http://www.theguardian.com/technology/2013/nov/11/apple-maps-google-iphone-users`, November 2013.



[30] Dirk Hähnel, Rudolph Triebel, Wolfram Burgard, and Sebastian Thrun. Map building with mobile robots in dynamic environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1557–1563, 2003.

[31] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50, 1988.

[32] Richard Hartley and René Vidal. Perspective nonrigid shape and motion recovery. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 276–289. Springer Verlag, Berlin, Heidelberg, 2008.

[33] Richard I. Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[34] Anders Heyden. Projective structure and motion from image sequences using subspace methods. In *Proceedings of the Scandinavian Conference on Image Analysis*, pages 963–968, 1997.

[35] Chen-Han Hsiao and Chieh-Chih Wang. Achieving undelayed initialization in monocular slam with generalized objects using velocity estimate-based classification. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2011.

[36] Anil K. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, NJ, USA, 1989.

[37] Yekeun Jeong, David Nistér, Drew Steedly, Richard Szeliski, and In-So Kweon. Pushing the envelope of modern methods for bundle adjustment. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 34, pages 1605–1617, August 2012.

[38] Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception & Psychophysics*, 14(2):201–211, 1973.

[39] H.M. Karara and L.P. Adams. *Non-topographic Photogrammetry*. Science and engineering series. American Society for Photogrammetry and Remote Sensing, 1989.

[40] L. Kontsevich, M. Kontsevich, and A. Shen. Two algorithms for reconstructing shapes. *Optoelectronics, Instrumentation and Data Processing*, 5:76–81, 1987.

[41] Karl Kraus, Josef Jansa, and Helmut Kager. *Photogrammetry*. Ferdinand Dümmlers Verlag, 1997.

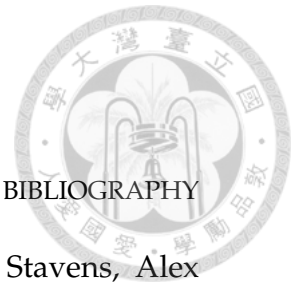
[42] Abhijit Kundu, K. Madhava Krishna, and C.V. Jawahar. Realtime multibody visual slam with a smoothly moving monocular camera. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2080–2087, November 2011.

[43] Henning Lategahn, Andreas Geiger, Bernd Kitt, and Christoph Stiller. Motion-without-structure: Real-time multipose optimization for accurate visual odometry. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, 2012.

[44] Thomas Lemaire, Cyrille Berger, Il-Kyun Jung, and Simon Lacroix. Vision-based slam: Stereo and monocular approaches. *International Journal of Computer Vision*, 74(3):343–364, 2007.

[45] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, II(2):164–168, 1944.

[46] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Sören Kammel, Zico J. Kolter, Dirk Langer, Oliver Pink, Vaughan



Pratt, Michael Sokolsky, Ganymed Stanek, David M. Stavens, Alex Teichman, Moritz Werling, and Sebastian Thrun. Towards fully autonomous driving: Systems and algorithms. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 163–168, June 2011.

[47] Ting Li, Vinutha Kallem, Dheeraj Singaraju, and René Vidal. Projective factorization of multiple rigid-body motions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–6, 2007.

[48] Kuen-Han Lin and Chieh-Chih Wang. Stereo-based simultaneous localization, mapping and moving object tracking. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3975–3980, 2010.

[49] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1150–1157, 1999.

[50] Simon Lucey and Jack Valmadre. General trajectory prior for non-rigid reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1394–1401, 2012.

[51] Yi Ma, Stefano Soatto, Jana Kosecka, and S. Shankar Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003.

[52] Donald W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *SIAM Journal on Applied Mathematics*, 11(2):431–441, 1963.

[53] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, 2004.

[54] Kemal Egemen Ozden, Konrad Schindler, and Luc Van Gool. Multibody structure-from-motion in practice. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 32(6), pages 1134–1141, Los Alamitos, CA, USA, 2010.

[55] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 3d reconstruction of a moving point from a series of 2d projections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 158–171. Springer Verlag, Berlin, Heidelberg, 2010.

[56] Kamisetty Ramamohan Rao and Ping Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[57] Konrad Schindler, David Suter, and Hanzi Wang. A model-selection framework for multibody structure-and-motion of image sequences. *International Journal of Computer Vision (IJCV)*, 79(2):159–177, 2008.

[58] D.A. Shulman and J. Aloimonos. *(Non)rigid Motion Interpretation: A Regularized Approach*. Number no. 1860 in CAR (Series). University of Maryland, 1987.

[59] Randall Smith, Matthew Self, and Peter Cheeseman. A stochastic map for uncertain spatial relationships. In *Proceedings of the 4th International Symposium on Robotics Research*, pages 467–474, Cambridge, MA, USA, 1988. MIT Press.

[60] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH Conference Proceedings*, pages 835–846, New York, NY, USA, 2006. ACM Press.

[61] Germany Statistical Federal Office. Verkehr, verkehrsunfälle, fachserie 8, reihe 7 (annual report on traffic accidents).

https://www.destatis.de/DE/Publikationen/Thematisch/TransportVerkehr/Verkehrsunfaelle/VerkehrsunfaelleJ.html, July 2014.

[62] Peter Sturm and Bill Triggs. A factorization based algorithm for multi-image projective structure and motion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 709–720. Springer Verlag, Berlin, Heidelberg, 1996.

[63] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer London, 2010.

[64] Sebastian Thrun. Leave the driving to the car, and reap benefits in safety and mobility. *The New York Times*, December 2011.

[65] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. Stanley: The robot that won the darpa grand challenge. *Journal of Field Robotics*, 23(9):661–692, 2006.

[66] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision (IJCV)*, 9(2):137–154, 1992.

[67] Phil Tresadern and Ian Reid. Articulated structure from motion by factorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[68] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment — a modern synthesis. In *Vision Algorithms: Theory and Practice*, volume 1883 of *Lecture Notes in Computer Science*, pages 298–372. Springer Berlin Heidelberg, 2000.

[69] Shimon Ullman. Maximizing rigidity: The incremental recovery of 3-d structure from rigid and and rubbery motion. *Perception*, 13:255–274, 1983.

[70] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M. N. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas M. Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt McNaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[71] René Vidal and Daniel Abretske. Nonrigid shape and motion from multiple perspective views. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 3952, pages 205–218. Springer Verlag, Berlin, Heidelberg, 2006.

[72] Chieh-Chih Wang, Charles Thorpe, and Sebastian Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded

urban areas. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 842–849, 2003.

[73] Chieh-Chih Wang, Charles Thorpe, Sebastian Thrun, Martial Hebert, and Hugh Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *International Journal of Robotics Research*, 26:889–916, 2007.

[74] World Health Organization (WHO). Global status report on road safety 2013. `http://www.who.int/violence_injury_prevention/road_safety_status/2013/en/`, March 2013.

[75] Wikipedia. Dashcam. `http://en.wikipedia.org/wiki/Dashcam`, 2014.

[76] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz. Multicore bundle adjustment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3057–3064, Washington, DC, USA, 2011.

[77] Jing Xiao, Jinxiang Chai, and Takeo Kanade. A closed-form solution to non-rigid shape and motion recovery. *International Journal of Computer Vision (IJCV)*, 67(2):233–246, April 2006.

[78] Jing Xiao and Takeo Kanade. Uncalibrated perspective reconstruction of deformable structures. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2005.

[79] Jingyu Yan and Marc Pollefeys. A factorization-based approach to articulated motion recovery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–821, 2005.

[80] Jingyu Yan and Marc Pollefeys. Automatic kinematic chain building from feature trajectories of articulated objects. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR), pages 712–719, 2006.

[81] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions Pattern Analysis Machine Intelligence (PAMI)*, 22(11):1330–1334, November 2000.

[82] Danping Zou and Ping Tan. Coslam: Collaborative visual slam in dynamic environments. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 35(2), pages 354–366, February 2013.

Document Log:

Manuscript Version 4.2 — 30 June 2015

Typeset by LaTeX — 22 July 2015

ANDREAS DOPFER

THE ROBOT PERCEPTION AND LEARNING LAB., DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION ENGINEERING, NATIONAL TAIWAN UNIVERSITY, NO.1, SEC. 4, ROOSEVELT RD., DA-AN DISTRICT, TAIPEI CITY, 106, TAIWAN, *Tel.* : (+886) 2-3366-4888 EXT.407

E-mail address: andi@pal.csie.ntu.edu.tw