ETH zürich

IGP
Institute of Geodesy and Photogrammetry

Drone Trajectory Reconstruction using Multi-Camera System

by

Jingtong Li

Master Thesis

in Geomatics Engineering

Chair of Photogrammetry and Remote Sensing

Institute of Geodesy and Photogrammetry

ETH, Zürich

Professorship:	Prof. Dr. Konrad Schindler
Supervision:	Dr. Cenek Albl
Date of Submission:	1st July, 2019

Spring Semester 2019

Acknowledgement

I would like to take this opportunity to express my sincere gratitude to everyone who supported me during this master thesis:

- **Prof. Dr. Konrad Schindler** for offering me the opportunity of working on this project as my master thesis

- **Dr. Cenek Albl** for the supervision of this master thesis and his continuous assistance. He has made a great contribution of project organization, design of the project pipeline, data acquisition and guidance of the implementation etc. I strongly benefit from our discussion and his rich experience in the field of computer vision.

- **Zhenzhong Su** for providing useful RTK data of drone using product of his own company.

- **Jesse Murray** for his academic cooperation and technical assistance through the whole project.

- **Jacky Liao** for processing drone detections and tracking.

- **Dorina Ismaili** for her novel idea on bundle adjustment.

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Drone Trajectory Reconstruction using Multi-Camera System

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Li	Jingtong

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zurich, 01.07.2019	Jingtong

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

Abstract

Unmanned aerial vehicles, commonly known as drones, have been developed for decades and recently have been massively applied in many different areas. An essential subject of drones is precise localization, which is in increasing demand and has been studied by many researchers. While traditional approaches have their limitations, many recent works attempt to recover drone trajectories with computer vision-based methods. This thesis aims at exploring the potential of reconstructing 3D drone trajectories observed from multiple stationary cameras. We take sequences of 2D drone detections from videos as input. Based on that, we temporally synchronize our cameras as a first step. A 9-point solver and two algorithms built upon this solver are implemented to simultaneously estimate time offsets between two cameras and their relative poses. This camera synchronization acts as a pre-processing step in our complete pipeline, which follows the incremental approach of multi-view reconstruction. Moreover, we present a spline representation for optimizing the reconstruction in bundle adjustment, which helps implicitly constrain drone trajectories and reduce large errors. Experiments on synthetic data and on real videos from outdoor drone flights demonstrate the effectiveness of our approach. Our reconstructed drone trajectories conform to the RTK ground truth with deviations under tens of centimeters.

Contents

List of Figures

List of Tables

1 Introduction

This chapter is an introduction to the main ideas covered in this thesis. It gives a brief overview of the purpose and challenges of the thesis. The structure of the thesis is introduced at the end of this chapter.

1.1 Motivation

In recent times unmanned aerial vehicles, also commonly known as drones, play a more and more important role in our society. The history of drones can be traced back in the early 1900s, where the innovation focused mainly on military purposes. Nowadays with the continued evolution in technologies, drones have become a common and powerful tool in many areas of our society. For instance, drones provide a cost effective alternative to planes and helicopters for aerial surveying and photogrammetry. They are flexible and are able to fly very close to ground and access to dangerous area with less utilization of human resources. Another example of drone application field is agriculture. Agricultural drones enable farmers to observe their fields from sky, which could help increase crop production and monitor crop growth. Besides, drones are being deployed in disaster relief, conservation of biodiversity, movie industry, aerial surveillance and many other scenarios.

With constantly increasing deployment of drones, techniques of their tracking and localization are in high demand and present a challenging task. Recent events of drones flying over airports, city centers, private or restricted territories underscore the threat of uncontrolled drone flights and the lack of proper surveillance. Other safety issues include unintentional collisions, malicious use and further security vulnerabilities. In fact, tracking and localizing drones could be a vital measure to counter those problems or prevent them from happening. Another reason for drone localization is navigation and guidance. Although onboard control devices are available in many situations, an external source for spatial information of drones can further improve reliability of a smooth flight. Under certain circumstances, reproducibility of the same trajectory is imperative for surveillance and mapping purposes, for which most of existing techniques are either expensive or not precise enough.

In contrast to the high desirability, few approaches exist that thoroughly handle drone localization and trajectory recovery. The combination of IMU and GPS is a conventional solution. Yet IMU devices do not have exceptionally good performance characteristics mainly because of bias accumulated in long term. And GPS signals can be unavailable or limited in terms of accuracy in vertical direction, especially for tasks such as autonomous landing, low altitude positioning or indoor applications. Since both IMU device and GPS receiver belong to active sensors, the potential of passive approaches should be also investigated. Therefore, this thesis focuses on utilizing a computer vision solution to the problem of drone trajectory reconstruction, in other words leveraging a multi-camera system

1.2 Problem Formulation

The practical scenario of this work consists of a flying drone and multiple cameras on the ground. The cameras are stationary and capture the drone from different views producing video sequences. The input data of this work are detections of the drone in each frame in form of single pixel coordinates, which can be obtained by arbitrary object tracking method. Based on these 2D detections, a method is presented for drone trajectory reconstruction up to a similarity transformation with the real physical model. This setup can be regarded as a simulation for real scenarios, e.g. surveillance cameras mounted around a power plant or an airport.

The primary challenges of this work are **video synchronization**, **camera pose estimation** and **trajectory reconstruction**, which are essentially correlated problems. Most of todays multi-view tracking algorithms rely on the assumption that video sequences are temporally synchronized. Normally perfect synchronicity of video streams can be achieved by hardware synchronization in laboratory or studio situations. Achieving synchronization in large-scale outdoor environments, however, can be difficult, expensive or even impossible. In this work, the objective is to enable setting up simpler and cheaper capture configuration, such as using smartphone or surveillance cameras in a wild outdoor environment. Recent works have shown that camera synchronization can be achieved solely by investigating the 2D trajectory detections from video sequences. Additionally, camera pose estimation is another problematic issue as there is no actual correspondence between detections in different cameras. A straightforward idea is to exploit backgrounds of the scene for camera poses, as cameras are static. Nonetheless, our preliminary experiments show that the traditional feature matching methods fail most of the time. Because backgrounds usually contain textureless contents or repetitive patterns, like sky or forest, which do not provide enough features or create difficulties for feature matching. Therefore in this work the camera pose estimation is accomplished also depending on 2D trajectory detections after they are temporally aligned. Eventually, one potential issue of conventional reconstruction methods is the assumption that a scene should be reconstructed as a set of independent 3D points. As the object we want to recover in this work is a trajectory, it should follow basic physical principles and obey certain geometric rules, e.g. with respect to smoothness or continuity. Thus a method is presented in this work for representing the 3D drone trajectory using a spline. This spline representation could implicitly constrain the shape of the trajectory with only a few control points.

In summary, following assumptions should be fulfilled for a practical application of the presented approach and will be discussed in detail in the following chapters:

- At least two cameras are deployed, which are stationary. They are preferably calibrated, but not necessarily.
- The video sequences containing drone flights should have spatial and temporary overlap. Videos can be unsynchronized because of different recording starts or shutters triggering, which causes a constant time shift. Frame rate should be known for each camera.
- The 2D detections of drone are allowed to have outliers. For each frame at most one detection should be given.
- In principle, the drone is allowed to occasionally fly outside the field of view of one or more cameras. Yet at least parts of video sequences from different cameras should overlap, e.g. recording same pieces of the drone trajectory.
- The temporal distances between two detections should be constant for each video sequence. In case of videos with different frame rates, detections could be rescaled (e.g. downsampling or interpolation).

1.3 Thesis Structure

The remainder of this thesis is organized as follows: Chapter 2 summarizes previous work on the topics of video synchronization, multi-view 3D reconstruction and bundle adjustment. Chapter 3 provides an overview of general concepts and principles exploited as foundations of this work. Chapter 4 presents the complete approach of drone trajectory reconstruction in detail. Experiments and results of the proposed methods on both synthetic and real data are presented in Chapter 5 and further discussed in Chapter 6. Finally, the thesis is concluded in Chapter 7.

2 Related Work

Trajectory reconstruction from multiple views has been investigated considerably in the past. This chapter reviews recent works that are most relevant to the context of this thesis.

2.1 Video Synchronization

Different strategies exist in terms of solving video synchronization in the computer vision area. Many approaches are based on image content analysis. For instance, Caspi et al., 2002 exploited temporal variations between image frames (such as changes in scene illumination) as cue for alignment. They suggested that folding spatial and temporal cues into a single alignment framework could deal with situations which are inherently ambiguous for traditional image-to-image alignment methods. Yan et al., 2004 detected space-time interest points and described their variations as a temporal distribution. The temporal shift can then be calculated through by assuming similar distributions in two video sequences. In Dai et al., 2006, an iterative algorithm is presented using 3D phase correlation based on a projective geometry constraint. A simplifying assumption has to be made that the centres of the cameras have to be relatively close to each other.

Another group of methods leverage epipolar geometry and object trajectory matching. A set of trajectories are detected in each video sequences by 2D tracker. These trajectories are then matched by seeking a spatial and temporal transformation. Caspi et al., 2006 applied RANSAC to search for matching trajectory pairs from a reduced set of all combinations of trajectories. Here the epipolar geometry is assumed to be provided. The method from Padua et al., 2008 extended the problem of pairwise synchronization into joint synchronization of N sequences. This required the estimation of a single N-dimensional line called *timeline*. Vo et al., 2016 did an exhaustive search on a discretized set of temporal offsets after a initial guess. Then the offset with the smallest epipolar geometry error is taken as the sub-frame alignment.

The most closely related previous works to this thesis are Noguchi et al., 2007, Nischt et al., 2009 and Albl et al., 2017, as they jointly perform video synchronization and estimate the two-view geometry based on image point trajectories. Noguchi et al., 2007 approximated local 2D image trajectory by a straight line and then estimated epipolar geometry or homography together with lags in shutter time using non-linear least square optimization. Nischt et al., 2009 extended this method by allowing estimation of difference in frame rate and approximating the local trajectory by fitting a spline. Both these approaches, however, formulated the problem as cost minimization, which needs a good initial guess and is solved using an iterative alternating algorithm. Thus they are only able to deal with small time shift, e.g. within 0.25s. In contrast, our method does not need a good initial estimate and is able to process large time offsets of several seconds, but with computationally lower cost than the exhaustive search. The method applied in this work is mainly inspired by Albl et al., 2017. In addition to their original work, a fixed-time algorithm for searching time shift is presented in this thesis as an alternative in case the iterative algorithm fails to converge or converge in large number of iterations.

2.2 Dynamic 3D Reconstruction

Synchronized multi-view system has been prevalent for dynamic scene reconstruction. Unlike conventional 3D reconstruction where sparse static feature points are recovered, many dynamic reconstruction methods focus on recovering moving objects at different times. Most of current approaches for trajectory recovery appeal to motion priors to constrain reconstructions. For instance, Avidan et al., 2000 proposed a method for linear and conical motion, where multiple non-coincidental projections of a point are reconstructed. Yuan et al., 2006 followed a similar idea and extended to reconstruction of curved and general planar trajectories. Another group of methods leveraged geometric constraints implicitly. Valmadre et al., 2012 considered the smoothness of trajectories and defined a novel reconstruction error by exploiting high-pass filters. Zhu et al., 2013 stressed sparsity priors of trajectory recovery by focusing on the reduced isometry property condition. They applied convolutional sparse coding to learn the trajectory basis matrix.

Apart from geometric motion priors, some methods investigated physical motion priors that could potentially find correct temporal alignment of image trajectories. Vo et al., 2016 introduced priors that favor trajectory motion with constant velocity or constant acceleration. They conjectured that the cumulative forces applied by mechanical systems are sparse and over a small duration of time, the true trajectory can be approximated by the path that minimizes the costs, defined by their motion priors. Rozantsev et al., 2017 included a more realistic motion model for drones (e.g. yaw, pitch and roll), which enables explicit recovery of parameters such as the control inputs given by pilot. This physical model is then integrated into bundle adjustment and favors trajectories that can be well explained by this model in intermediate steps.

There are also works that do not follow the idea of motion prior constraints. For instance, Sinha et al., 2010 accomplished dynamic 3D reconstruction using silhouettes correspondences. Zheng et al., 2015 addressed the spatial and temporal aspects independently and recovered the unknown structure without sequencing information across video sequences. Nonetheless, these approaches are essentially pose-oriented, especially human poses, which barely work for tiny objects in outdoor scenes.

2.3 Bundle Adjustment

Conventional bundle adjustment (Triggs et al., 1999) often acts as a non-linear least square optimization step in 3D reconstruction, usually solved by some variations of Gauss-Newton algorithms such as the Levenberg-Marquardt algorithm. In the past years, many variants are proposed seeking for improvements in different aspects. Some approaches aim to reduce the complexity of bundle adjustment. Lourakis et al., 2009 published a software package for realizing generic bundle adjustment with high efficiency and flexibility regarding parameterization. They reduced the factorization time by a large constant factor by means of the Schur complement method. Agarwal et al., 2010 explored the use of conjugate gradients for calculating the Newton step and showed that this truncated Newton method offered state of the art performance for large-scale scenes. Wu, 2013 further utilized preconditioned conjugate gradient and showed that incremental Structure from Motion (SfM) requires only O(n) time w.r.t the number of cameras including bundle adjustment.

Another group of works modified the traditional bundle adjustment in case when regularization is incorporated. Cohen et al., 2012 proposed a method that imposed a set of structural constraints derived from the symmetry relations. Gong et al., 2015 exploited inaccurate parameters constrained in a range and presented a bound constrained bundle adjustment.

In this work, a variant of bundle adjustment is studied where parameters of a trajectory do not refer to 3D coordinates of sample points. As a trajectory can be approximated by a spline in presence of noises, spline coefficients are then optimized in the bundle adjustment, which can then recover the trajectory easily by resampling points at arbitrary positions.

3 Theoretical Foundations

This chapter gives an overview of basic concepts and principles outlined in previous works that are relevant to the subject of drone trajectory reconstruction. Special attention is paid to the section 3.2, where the standard 3D reconstruction pipeline is introduced in steps. The book by Hartley et al., 2003 is highly recommended for readers who want to have a thorough review of the underlying topics.

3.1 Object Tracking

Object tracking is not the main component of this work, as its results (i.e. 2D image trajectories of drone) are considered as input data in this work. Thus in this section the applied method is briefly introduced, which consists of background subtraction and mean-shift algorithm.

Background subtraction is a widely used technique for extracting a foreground mask, i.e. a binary image containing the pixels belonging to moving objects in the scene, by using a stationary camera. Here the method by Zivkovic et al., 2006 is applied, which is based on a non-parametric adaptive density estimation. After background subtraction only pixels considered as foreground are candidates containing the desired drone. Here at a first step, the initial location of drone has to be provided and thus the drone can be represented by a pixel histogram. Then the mean shift algorithm, which is a common tool for locating the maxima of a density function proposed by Fukunaga et al., 1975, could locate the updated histogram by matching with previous one in several steps. Finally by exploiting mean-shift over a video sequence, the drone in each frame is detected as single pixel coordinates.

3.2 Reconstruction Pipeline

3.2.1 Camera calibration

Camera calibration is the process of estimating camera parameters, which consist of intrinsic and extrinsic parameters. Intrinsic parameters deal with camera's internal characteristics, while extrinsic parameters describe the camera position

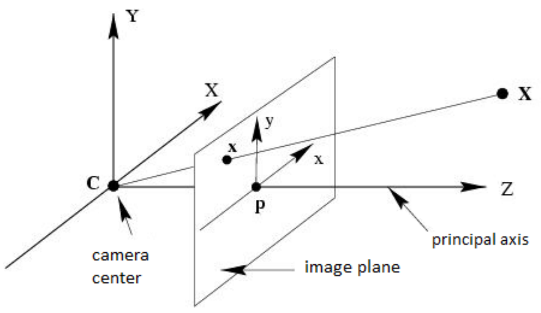


Figure 3.1: Perspective camera model

and orientation in the real world. In case of a perspective camera model (see Fig. 3.1), camera parameters yield the following equations.

$$\lambda x = PX \tag{3.1}$$

$$P = K[R|t] \tag{3.2}$$

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Here X denotes a point in space and x denotes its projection on the image plane. λ is the perspective depth and P is the camera matrix, which has the form of 3×4, as X and x are in homogenous coordinates. P can be decomposed into a calibration matrix K (i.e. intrinsic parameters) and a rotation R and a translation t (i.e. extrinsic parameters). Among entries in K, f_x and f_y denote focal length in pixels, s is the skew coefficient which is non-zero if image axes are not perpendicular, and c_x, c_y are the pixel coordinates of the optical center. As many models exist for correcting radial distortions, the most commonly used one is introduced by Brown (1966) described in Eq. 3.4, which is a decentering model and radial distortions are described as coefficients of these polynomial equations.

$$\begin{aligned} x_{distorted} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{distorted} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{3.4}$$

In this work, camera calibration refers to the determination of intrinsic parameters using standard method proposed by Zhang (2000) with a chessboard. Furthermore, based on the current manufacturing techniques, it is assumed that skew is zero for all cameras and the first two coefficients of Eq.3.4 are sufficient for correcting radial distortions.

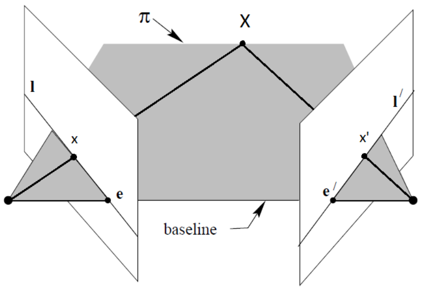


Figure 3.2: Epipolar geometry

3.2.2 Epipolar geometry

Epipolar geometry describes the relation between two camera views observing the same objects, which enables the determination of the third dimension (i.e. depth along the ray) and thus the 3D reconstruction. The essence of epipolar geometry is the coplanarity constraint, which is illustrated by the basic setup in Fig. 3.2. Here the baseline, which is the line connecting the two camera centers, intersects each image plane at the epipoles e and e'. Any plane π containing the baseline is an epipolar plane, and intersects the image planes in corresponding epipolar lines l and l' (Hartley et al. (2003)).

Fundamental Matrix The fundamental matrix is a 3×3 rank 2 matrix representing epipolar geometry algebraically and satisfies the condition in Eq. 3.5 for any pair of corresponding points $x \leftrightarrow x'$ in two images. This means any point x' in the second image matching the point x must lie on the epipolar line l', which is described by $l' = Fx$ and valid vice versa. Thus Eq. 3.5 characterizes the fundamental matrix without reference to the camera parameters, i.e. solely from corresponding image points.

$$x'^{\top} F x = 0 \tag{3.5}$$

$$\begin{gathered}
[x' \quad y' \quad 1] \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \\
\Downarrow \\
xx'f_1 + yx'f_2 + x'f_3 + xy'f_4 + yy'f_5 + y'f_6 + xf_7 + yf_8 + 1 = 0 \\
\Downarrow \\
[xx' \quad yx' \quad x' \quad xy' \quad yy' \quad y' \quad x \quad y \quad 1]\, f = 0
\end{gathered} \tag{3.6}$$

For the computation of the fundamental matrix, the most commonly used model is the Eight-point algorithm from Hartley, 1995. Concrete steps of this algorithm are not listed in this work, as they can be easily found in the literature. Here the most essential part of this algorithm is presented, as each point correspondence yields the linear equation derived in Eq. 3.6, where f is the flattened vector containing the nine unknown entries of F. For eight point correspondences the resulting eight equations can be combined to the linear system in Eq. 3.7

$$Mf = 0 \tag{3.7}$$

with M being a 8×9 matrix. The least-squares solution for f is the singular vector corresponding to the smallest singular value of M, that is, the last column of V in the SVD decomposition $M = UDV^T$. An important property of the fundamental matrix is that it is singular and has rank 2. This singularity of F could be enforced by decomposing F using SVD into $F = UDV^T$, setting $D(3,3) = 0$ and composing F again as $F = UDV^T$, i.e. letting the smallest singular value of F equals zero.

As we assume that raw input detections include noise and outliers, a better solution could be incorporating this eight point solution into a RANSAC framework (see Section 3.3). As for the error measure used within RANSAC, the Sampson error is often applied as a first-order approximation to the geometric error (Hartley et al. (2003)).

$$\text{Sampson error} = \frac{\left(\mathbf{x}'^\top \mathbf{F}\mathbf{x}\right)^2}{(\mathrm{F}\mathbf{x})_1^2 + (\mathrm{F}\mathbf{x})_2^2 + (\mathrm{F}^\top\mathbf{x}')_1^2 + (\mathrm{F}^\top\mathbf{x}')_2^2} \tag{3.8}$$

Essential Matrix Essential matrix can be seen as the specialization of the fundamental matrix to the case of known camera intrinsic parameters. It expresses the relation between image coordinates in calibrated camera coordinate system that can be obtained as $\hat{x} = K^{-1}x$, with known calibration matrix K. The following equations define the essential matrix and show its relation with the fundamental matrix.

$$\hat{x}'^\top E\hat{x} = 0 \tag{3.9}$$

$$E = K'^\top FK \tag{3.10}$$

By definition, essential matrix can be directly estimated using the Eight-point algorithm. Another solution, proposed by Nistér, 2004, estimates E using only 5 point correspondences (E has 5 degrees of freedom). Similar to F, the rank deficiency as well as the degrees of freedom constraint has to be enforced, which is typically done by setting the first two singular values of E identical and letting the third singular value equals zero. Essential matrix is very useful for calibrated reconstruction, as the relative camera pose can be extracted from it. Assuming the first camera $P = [I \mid 0]$, the rotation and translation of the second camera can be retrieved from E up to a four-fold ambiguity, i.e. four possible solutions. The correct solution can be determined by checking that reconstructed points should be in front of both cameras.

3.2.3 Focal length estimation

The motivation of camera focal length estimation comes from the problem of Euclidean reconstruction (up to a similarity transformation from the true reconstruction) from two cameras with unknown intrinsic parameters. It has been proved that Euclidean reconstruction from uncalibrated cameras is not possible (Hartley et al., 1992), as the best possible result is projective reconstruction. Yet it is theoretically possible to retrieve focal length of cameras, assuming square pixels (no skew and unit aspect ratio) and provided an estimate of the principal point (e.g. at image center).

A simple implementation following this idea is presented by Bougnoux (1998), which relies on the fundamental matrix and can be summarized by the following equation:

$$f = \sqrt{-\frac{\mathbf{p}'^T \left[\mathbf{e}'\right]_\times \tilde{\mathbf{I}}\mathbf{F}\mathbf{p}\mathbf{p}^T\mathbf{F}^T\mathbf{p}'}{\mathbf{p}'^T \left[\mathbf{e}'\right]_\times \tilde{\mathbf{I}}\mathbf{F}\tilde{\mathbf{I}}\mathbf{F}^T\mathbf{p}'}} \tag{3.11}$$

where f is the focal length of the first camera, p and p' the principal points, e and e' the epipoles and $\tilde{I}$ the diagonal matrix diag(1,1,0). The focal length of the second camera f' is obtained by interchanging the roles of the two cameras.

An improved method is later proposed by Hartley et al., 2002, which formulated the problem as optimization of a cost function that penalizes unrealistic focal length estimates and optimized principal points far away from nominal positions. This method is tested in the preliminary stage of this work and yet verified as unreliable for our type of data. Because it is quite sensitive to the assumed position of the principal point as well as the computed fundamental matrix and can not guarantee a convergence to feasible solutions each time. Thus the focal length estimation is not applied to our experiments on real datasets. Instead, we presume focal lengths that deviate 20% from the calibrated values as a practical estimate for the experiments of reconstruction using uncalibrated cameras.

3.2.4 Triangulation

The term triangulation is understood as the recovery of a 3D point with known projections from multiple camera views. In case of two camera views, a mathematical expression of this geometric relation is shown by the following equations:

$$\begin{aligned} x &= PX \\ x' &= P'X \end{aligned} \tag{3.12}$$

where X denotes an unknown 3D point, P, P' are the camera matrices and x, x' are corresponding 2D projections of this point.

The most common solution is the linear triangulation (see Eq. 3.13). In this method each camera view gives rise to two equations on the three entries of X. Combining these four equations, a overdetermined linear equation system is obtained and can be solved in a least square sense, often by SVD.

$$\begin{gathered} x = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad x' = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} \quad P = \begin{bmatrix} p_1^T \\ p_2^T \\ p_3^T \end{bmatrix} \quad P' = \begin{bmatrix} p_1'^T \\ p_2'^T \\ p_3'^T \end{bmatrix} \\ \Downarrow \\ AX = 0 \qquad \text{with } A = \begin{bmatrix} up_3^T - p_1^T \\ vp_3^T - p_2^T \\ u'p_3'^T - p_1'^T \\ v'p_3'^T - p_2'^T \end{bmatrix} \end{gathered} \tag{3.13}$$

The minimized algebraic error in this method, however, is not geometrically meaningful. A better solution is proposed by Hartley et al., 1997, which is referred to the optimal method of triangulation. The key idea behind it is to find the correct correspondence $\hat{x} \leftrightarrow \hat{x}'$ lying close to the measured correspondence $x \leftrightarrow x'$ and subject strictly to the epipolar constraint $\hat{x}'^\top F\hat{x} = 0$. The closeness to the measured correspondence is expressed as the optimization of the following function:

$$d(\mathbf{x}, \hat{\mathbf{x}})^2 + d\left(\mathbf{x}', \hat{\mathbf{x}}'\right)^2 \tag{3.14}$$

where $d(*, *)$ denotes Euclidean distance. After the correct correspondence $\hat{x} \leftrightarrow \hat{x}'$ is found, the 3D point X is then computed simply using linear triangulation. This non-linear triangulation method is applied in this work for the initial pair of cameras with fundamental matrix. For each additional camera the linear triangulation is exploited.

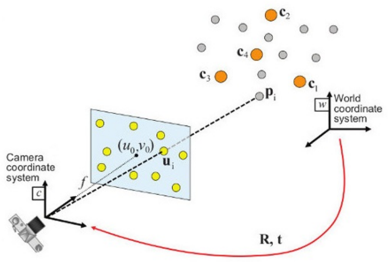


Figure 3.3: Perspective-n-Point model [1]

3.2.5 Perspective-n-Point

Perspective-n-Point (or PnP) is the problem of estimating the pose of a calibrated camera from a set of n 3D$\leftrightarrow$2D point correspondences (see Fig. 3.3). The term camera pose is equivalent to camera extrinsic parameters, which involves rotation and translation w.r.t the world coordinate system (see Eq. 3.2) and has 6 degrees of freedom.

As a camera pose has 6 DOF (3 rotation and 3 translation) and each correspondences provides two equations, the minimum set of correspondences required to this problem is n=3, which is therefore often denoted as P3P. P3P yields up to four feasible solutions, which leads to the need for a fourth correspondence to remove ambiguity. In the context of this work, the method proposed by Lepetit et al., 2009 is applied, which is commonly known as EP3P solution. Built upon this, the RANSAC algorithm (see Section 3.3) is used to make the approach resistant to outliers. A global optimization that minimizes sum of reprojection errors is also imposed on all the inliers.

3.2.6 Bundle adjustment

Bundle adjustment often refers to the process of simultaneous refinement of structure parameters (e.g. 3D object points) and camera parameters. It is almost always applied as a last step of feature-based 3D reconstruction and essentially equivalent to the Maximum Likelihood Estimator assuming zero-mean Gaussian image errors.

A typical implementation of bundle adjustment is based on minimizing the sum

[1]Source: OpenCV library

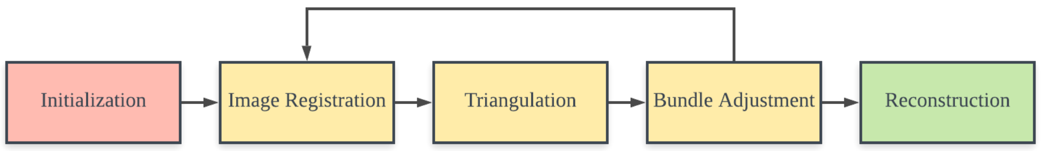


Figure 3.4: A basic workflow of incremental Structure from Motion

of squared reprojection errors as shown in Eq. 3.15

$$\arg\min_{P_i,\mathbf{X}_j} \sum_{i=1}^{m} \sum_{j=1}^{n} \|\mathbf{x}_{ij} - P_i\mathbf{X}_j\|^2 \tag{3.15}$$

where P_i denotes the individual camera matrix, $\mathbf{X}_j$ the reconstructed object point and $\mathbf{x}_{ij}$ the image projection of point $\mathbf{X}_j$ on the camera P_i. The minimization in Eq. 3.15 is achieved using nonlinear least square algorithms, which require a good initialization and can be computationally very expensive because of the number of parameters involved. A general solution is the Levenberg-Marquardt algorithm (LM) by Marquardt, 1963, which is proved to be very efficient by leveraging the sparsity character of the Jacobian matrix.

3.2.7 Structure from Motion

Structure from Motion (SfM) is a general approach in computer vision for jointly estimating 3D scene (structure) and cameras from a set of 2D images (motion). Basically, it includes all the above outlined techniques in Section 3.2, as well as some additional steps. As there are different paradigms for SfM, a widely applied scheme is the incremental reconstruction, whose basic workflow is illustrated in Fig. 3.4. Other strategies include global (e.g. Brand et al., 2004) or hybrid (e.g. Cui et al., 2017) structure from motion, which could potentially improve efficiency at the price of robustness and accuracy.

In the initialization step the motion of the first two images is computed using fundamental matrix or essential matrix. An initial structure is then obtained by triangulating corresponding points. Then for each additional view, the camera is registered to the structure by solving the PnP problem using the known 3D points visible in its image. After the estimation of camera projection matrix, new 3D points can be added to the structure by triangulation. This is followed by a bundle adjustment optimizing the current structure and cameras. Bundle adjustment is also applied in the final step when all images are registered and the structure is complete, to find the optimal structure and camera parameters that

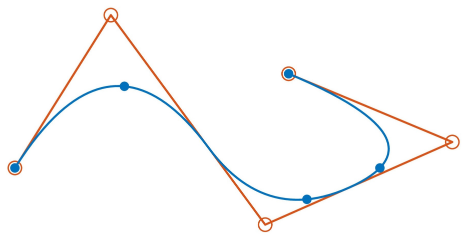


Figure 3.5: An example of B-spline [2]. Control points are in red and knot points in blue

minimize the total reprojection error.

3.3 RANSAC Algorithm

Random sample consensus (RANSAC) is a robust model fitting method that estimates a mathematical model from a set of observed data contaminated by outliers. It assumes that the data consist of inliers, whose distribution can be explained by model parameters, and outliers that do not fit to this distribution. As RANSAC is a non-deterministic algorithm, the first step is random sampling of a small subset of individuals. Then a model can be established based on this subset of data. Any other data that are close to this model within a threshold are considered as inliers. This process is then repeated and the finial model is the one with most inliers. A modification of this basic setup is the local optimized RANSAC (LO-RANSAC) proposed by Chum et al., 2003, where eventually an optimization (e.g. least-square) is imposed on the complete set of inliers. The RANSAC algorithm is widely applied in computer vision, e.g. estimating fundamental matrix and solving the PnP problem.

3.4 Spline Approximation

By definition, a spline is a piecewise polynomial (parametric) curve. Spline functions are used for interpolation given sample data points and can represent many different types of curves in space. Spline approximation has been proved to be a powerful tool for curve fitting as it can approximate complex shapes with high accuracy and less computation.

In this work, the cubic B-spline representation is applied to approximate drone trajectories. A B-spline curve is a generalization of Bézier curve and is defined by

[2]Source: bsplines.org

$n+1$ control points, $m+1$ knots and base functions of degree p, where m, n and p are related by the condition $(m+1) = (n+1)+(p+1)$. A mathematical definition of B-spline is given in the following equation:

$$\mathbf{C}(t) = \sum_{i=0}^{n} N_{i,p}(t)\mathbf{P}_i \tag{3.16}$$

where $T = (0, \cdots, 0, t_{p+1}, \cdots, t_{m-p+1}, 1, \cdots, 1)$ is the knot vector, $\mathbf{P}_i$'s are the control points (red points in Fig. 3.5) and $N_{i,p}(t)$'s are the B-spline base functions of degree p (p=3 in case of cubic spline). The point on the curve that corresponds to a knot t_j, $C(t_j)$, is referred to as a knot point (blue points in Fig. 3.5). A cubic B-spline with uniform knot vector can be thus constructed simply given coordinates of all the control points.

4 Methodology

This chapter provides the underlying methodology of this work in depth. In section 4.1 a method is implemented for synchronizing two cameras considering their relative geometry. In section 4.2 the pipeline of drone trajectory reconstruction is presented after cameras are globally synchronized.

4.1 Synchronization of Two Cameras

A method for simultaneous estimation of synchronization of two cameras and the fundamental matrix using only image correspondences from Albl et al., 2017 is presented in this section. A *quasi-minimal solver* introduced in section 4.1.2 gives the time shift estimate between two cameras using a small set of correspondences. The algorithms in section 4.1.3 are built upon this solver enabling estimation of large time shifts.

4.1.1 Relation between two unsynchronized cameras

The movement of a drone can be considered as a trajectory of a 3D point in space representing the drone center. The coordinates of this point can be described as

$$X(t) = [X_1(t), X_2(t), X_3(t), 1]^T \tag{4.1}$$

where t denotes time. Projecting $X(t)$ into image planes of two distinct cameras produces two 2D image trajectories. As the camera captures discrete frames of $X(t)$, two sequences of samples can be obtained from the two cameras:

$$x_i = \pi(X(t_i)), \quad i = 1, 2, \ldots n \tag{4.2}$$

$$x'_j = \pi'(X(t'_j)), \quad j = 1, 2, \ldots n' \tag{4.3}$$

where i and j are frame indices, x_i and x'_j are 2D image points of drone and π and π' are projection functions of the two cameras, respectively.

Generally, there is no correspondence direct from input detections regarding x_i and x'_j that come from the same 3D point X, i.e. for $i = j$, x_i and x'_j do not represent

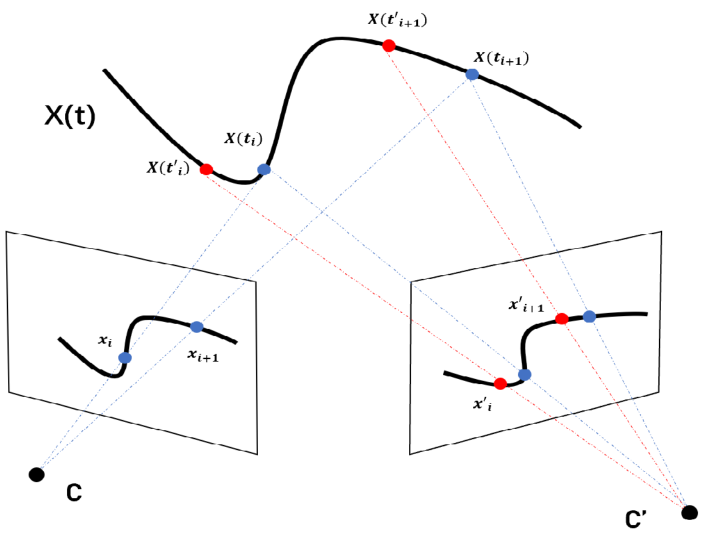


Figure 4.1: A moving point is captured in two cameras, yet there is no correct correspondence. On the right image, red points are measured and blue points are the true correspondences

the projection of the same 3D point. Two main sources of this asynchronization can be assumed. First, cameras may not start capturing video at the same time, which introduces a constant time shift between video sequences. Second, cameras may have different frame rates or inaccurate internal clocking, which leads to different time scales. The result of these two factors is a linear relationship between the times at which frames are captured

$$\mathbf{j}(i) = \alpha i + \beta \tag{4.4}$$

where $\alpha \in \mathbb{R}$ denotes the time scaling and $\beta \in \mathbb{R}$ denotes the time shift in units of number of frames. Note that Eq. 4.4 is an integer-to-real mapping. Thus $\mathbf{j}(i)$ may not be an integer but $x'(\mathbf{j}(i))$ can be approximated utilizing interpolation given image samples $x'_j,\ j = 1, 2, \dots n'$. In this way, a set of 2D point correspondences could be established

$$x_i \longleftrightarrow x'(\alpha i + \beta) \tag{4.5}$$

Solving synchronization between cameras thus reduces to the estimation of these two parameters (α, β) using correspondences in Eq. 4.5. In practice, the time scaling α is usually known or can be simply calculated. On the other hand, the time shift β is hardly known precisely unless in strictly controlled laboratory setups. Therefore in the rest of this work, it is assumed that $\alpha = 1$, which should be true when using same cameras or can be achieved by proper downsampling of video sequences. The focus of synchronization is then the time shift β.

4.1.2 A quasi-minimal solver of epipolar geometry

According to the 2D correspondences in Eq. 4.5 and the assumption of $\alpha = 1$, the general epipolar constraint of two cameras (see Eq. 3.5) has to be modified into the following equation:

$$x'(i+\beta)^T F x_i = 0 \tag{4.6}$$

In practice, projected image trajectories of drone x_i and $x'(i+\beta)$ are complicated curves. Therefore an approximation is exploited to reduce the problem into a first-order polynomial system, i.e. a linear system. This means that the real point $x'(i+\beta)$ is approximated by leveraging the sample point x'_i and a vector $v(d)$ as

$$v(d) = \frac{x'_{i+d} - x'_i}{d} \tag{4.7}$$

$$x'(i+\beta) = x'_i + v_i(d) \cdot \beta \tag{4.8}$$

where $v(d)$ is an approximation of the 2D trajectory over the next d samples, where d is called interpolation distance. The idea of this approximation is based on a simple motion model, where x'_i is the initial position, $v_i(d)$ is the velocity vector and β is the duration time.

Based on this model the epipolar geometry in Eq. 4.6 can be solved by the following equation system

$$\begin{gathered}
[x' + v_x\beta \quad y' + v_y\beta \quad 1] \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \\
\Downarrow \\
\begin{array}{c} x(x' + v_x\beta)f_1 + y(x' + v_x\beta)f_2 + (x' + v_x\beta)f_3 \\ +x(y' + v_y\beta)f_4 + y(y' + v_y\beta)f_5 + (y' + v_y\beta)f_6 \\ +xf_7 + yf_8 + 1 \end{array} = 0 \\
\Downarrow \\
\begin{array}{c} [xx' \quad yx' \quad x' \quad xy' \quad yy' \quad y' \quad x \quad y \quad 1]\, f \\ +\beta\, [xv_x \quad yv_x \quad v_x \quad xv_y \quad yv_y \quad v_y \quad 0 \quad 0 \quad 0]\, f \end{array} = 0
\end{gathered} \tag{4.9}$$

As an additional parameter β needs to be solved, 9 point correspondences are necessary to solve the Eq. 4.9, resulting to the following linear system

$$(M_1 + \beta M_2)\, f = 0 \tag{4.10}$$

where M_1 and M_2 are 9×9 matrices from correspondences and approximated tangent vectors. The Eq. 4.10 is a generalized eigenvalue problem, which can be solved efficiently using existing numerical algorithms. As three columns of M_2 contain only zeros, M_2 has a rank of six and three from nine eigenvalues should be zero. Thus there are up to six possible solutions where eigenvalues correspond to time shifts and eigenvectors are corresponding fundamental matrices. Similar to the Eight-point algorithm, the resulting fundamental matrix does not necessarily has the rank 2, which can be enforced by letting its third singular value be zero using SVD.

This approach for simultaneous estimation of camera synchronization and epipolar geometry is originated from Albl et al. (2017). It is referred to a *quasi-minimal solver* in this work, as theoretically the actual minimal required number of correspondences is eight (seven for the fundamental matrix because of the constraint $det(F) = 0$ and one for the time shift β).

As the solver is computationally efficient relying on a small set of correspondences, it can be embedded into a RANSAC framework for robustness against noise and outliers. In terms of synchronization using image trajectories, noise and outliers mainly come from two sources. First, input data of image trajectories may contain misdetections. In practice, misdetections are inevitable and happen more frequently when the drone flies with a large velocity or the appearance size of the drone varies on a large scale. Second, the assumed simple motion model has its limitations. This model is intended to simulate motion with constant velocity in a local range. Points with significant changes of velocity or on the non-linear pieces of trajectory could be possibly considered as outliers. The usage of RANSAC can avoid these points and select by chance the parts that are straight and linear. Another reason for exploiting RANSAC is the ambiguity of results from the solver. Up to six possible eigenvalues (i.e. non-complex number) could be obtained from Eq. 4.10. Only one of them could be a feasible estimate of the time shift β, which could be determined by RANSAC.

This quasi-minimal solver together with RANSAC is tested on synthetic data, presented in section 5.1.2. Evaluation shows that the interpolation distance d is an important parameter for estimating different lengths of time shift. This leads to developing further algorithms that exploit the solver as a kernel and estimate large time offsets.

4.1.3 Iterative algorithm and fixed-time algorithm

In this section two algorithms are presented built upon the quasi-minimal solver introduced in section 4.1.2. Both algorithms are aimed at improving the performance and extending the applicability of the solver, mainly based on the performance of the solver on synthetic data in section 5.1.2. The common property of both algorithms is the idea of varying interpolation distances d, while they distinguish themselves from each other by searching over d in different manners. For both algorithms, once β is estimated, the fundamental matrix between the two cameras is then also determined according to Eq 4.10.

Algorithm 1 Iterative algorithm

```
Input:
    Image trajectories from two camera $s$, $s'$, maximal exponent $p_{max}$
Output:
    Time shift $\beta$, fundamental matrix $F$
 1: Let $ratio_{max}$=0, $\beta = 0$, $p = 0$
 2: while $p <= p_{max}$ do
 3:     Apply quasi-minimal solver with $d = 2^p$, obtain $\beta_1$, $F_1$, $ratio_1$
 4:     Apply quasi-minimal solver with $d = -2^p$, obtain $\beta_2$, $F_2$, $ratio_2$
 5:     if $ratio_1 > ratio_2$ then
 6:         $\beta_{temp} = \beta_1$, $F_{temp} = F_1$, $ratio_{temp} = ratio_1$
 7:     else
 8:         $\beta_{temp} = \beta_2$, $F_{temp} = F_2$, $ratio_{temp} = ratio_2$
 9:     end if
10:     if $ratio_{temp} > ratio_{max}$ then
11:         Update $s'$ according to $\beta_{temp}$
12:         $\beta = \beta + \beta_{temp}$, $F = F_{temp}$
13:         $ratio_{max} = ratio_{temp}$, $p = 0$
14:     else
15:         $p = p + 1$
16:     end if
17: end while
18: return $\beta$, $F$;
```

Iterative algorithm This iterative algorithm is firstly introduced in the previous work from Albl et al. (2017). The basic principle is to estimate the time offset of two cameras step by step. Specifically, after a time offset β is estimated, the two image sequences will then be updated by shifting the one to the other with β frames. This process is then repeated until the two sequences from the two cameras are considered optimally aligned. Evaluation of the quasi-minimal solver shows that the ratio of inliers obtained from RANSAC gives a good hint of how close the estimated β to the real-time shift. Thus the problem of estimating β can be seen as optimization with the inlier ratio being criteria. One prerequisite of this iterative method is that the direction of β in each iteration should be correct resulting in better and not worse synchronization. This condition is fulfilled as the β computed far away from the optimum, though not precise, often provides a good indicator of the direction towards final synchronization. Moreover, using large interpolation distance d yields increasingly better estimates of large β, which are not larger than the actual time offset. So the possible candidates of d should be designed properly for different scales of time offsets.

The analysis mentioned above leads to the construction of the iterative algorithm. In the beginning, the search interval of d should be defined. A feasible choice is the exponential numbers, i.e. $d \in \{\pm 2^0, \pm 2^1, \cdots, \pm 2^m\}$. At each iteration, the β and the fundamental matrix F are estimated trying different d in the search interval. Once a model has a higher ratio of inliers than the previous iteration, re-align the two cameras accordingly and repeat the process until going overall all possible d the ratio of inliers still won't increase anymore.

Algorithm 2 Fixed-time algorithm

```
Input:
    Image trajectories from two camera $s, s'$, step length $l$, number of steps $k$,
    interpolation distance for the second stage $d_2$
Output:
    Time shift $\beta$, fundamental matrix $F$
 1: Let $ratio_{max}$=0, $\beta_{stage1} = 0$
 2: for each $d \in [-kl, -(k-1)l, \cdots, -1, 1, \cdots, (k-1)l, kl]$ do
 3:     Apply quasi-minimal solver with $d$, obtain $\beta_{temp}$, $F_{temp}$, $ratio_{temp}$
 4:     if $ratio_{temp} > ratio_{max}$ then
 5:         $\beta_{stage1} = \beta_{temp}$
 6:         $ratio_{max} = ratio_{temp}$
 7:     end if
 8: end for
 9: Update $s'$ according to $\beta_{stage1}$
10: Apply quasi-minimal solver with $d = d_2$, obtain $\beta_1$, $F_1$, $ratio_1$
11: Apply quasi-minimal solver with $d = -d_2$, obtain $\beta_2$, $F_2$, $ratio_2$
12: if $ratio_1 > ratio_2$ then
13:     $\beta_{stage2} = \beta_1$, $F = F_1$
14: else
15:     $\beta_{stage2} = \beta_2$, $F = F_2$
16: end if
17: $\beta = \beta_{stage1} + \beta_{stage2}$
18: return $\beta$, $F$;
```

Fixed-time algorithm One potential drawback of the iterative algorithm is the large computational burden. In the worst case, all candidates of d in the search interval must be evaluated once at each iteration. This significantly increases the necessary computation, especially when the estimated time shift is small at each iteration resulting in a large number of iterations. This fixed-time algorithm provides an alternative to synchronize two cameras in a straightforward manner and consists of two stages. The first stage includes the estimation of β searching over different d in large and linear intervals defined by step length l and number of steps k (e.g.[-40,-20,-1,1,20,40] for $l = 20$,$k = 2$). The β estimated with the highest ratio of inliers will be considered as the current time offset and will be applied to roughly synchronize the two cameras. In the second stage, the interpolation distance d is set to a small fixed integer (e.g. $d = 1$). Both positive and negative values of this d will be applied to the result from the first stage. Again from both results, the one with a higher inlier ratio will be accepted and helps to refine the estimated β from the first stage. In this way, the computational cost is reduced and known in advance.

4.2 Reconstruction using Synchronized Cameras

Camera synchronization is the pre-processing step in our reconstruction pipeline and the initial estimates of time shifts will be later optimized in bundle adjustment. As many approaches exist for multi-view reconstruction, the reconstruction pipeline in this work follows the incremental principle with some modifications.

4.2.1 Incremental reconstruction

Currently incremental reconstruction is the standard approach for Structure from Motion that adds on one camera at a time to grow the scene. In this work, our reconstruction pipeline is also based on this idea, as shown in Fig 4.2. As most of these steps are introduced in section 3.2, basically differences and improvements in our work will be pointed out in this section.

Firstly, given detected image trajectories from videos, cameras need to be temporally synchronized. The aforementioned sections already explain this in detail and provide algorithms for the case of two cameras. In case of n cameras, where $n \geq 3$, we first compute the time shift between each pair of cameras as $\beta_{i,j}$. In order to verify a global consistency over all cameras, the following equation should be fulfilled

$$\beta_{1,2} + \beta_{2,3} + \cdots + \beta_{n-1,n} + \beta_{n,1} < \beta_{thres} \tag{4.11}$$

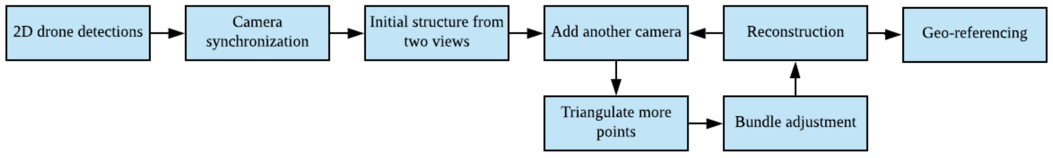


Figure 4.2: Reconstruction workflow

where β_{thres} is a threshold only allowing a weak inconsistency. In case the condition in 4.11 is not fulfilled, entries of the summation on the left side will be re-computed one at a time and each time this condition will be verified until satisfied or until a maximal number of iterations is reached. Once all cameras are consistently synchronized, a global timeline could be established. So each drone detection from each camera will be assigned to a global timestamp so that point correspondences over cameras are produced straightforwardly. Another modification of our pipeline is the bundle adjustment optimization, which will be discussed in section 4.2.2 and 4.2.3 in detail.

Eventually, the reconstructed drone trajectory is known to differ from the true trajectory by a Euclidean (similarity) transformation with uniform scaling when using calibrated cameras. In this work, we aim to perform geo-referencing by estimating this transformation for real data experiments where ground truth of drone locations are provided, e.g. RTK data. In order to obtain a transformation as precise as possible, we densely interpolated points from our reconstructed trajectory resulting in many different alignments with the ground truth. We estimated transformations for all possible alignments and selected the one with the smallest mean position error.

As the reconstructed trajectory usually has a higher measurement frequency than the ground truth data, the alignment to the real trajectory can be ambiguous up to the temporal resolution of the ground truth. Thus we estimate multiple possible transformations and choose the one with the smallest mean error.

4.2.2 Spline representation for drone trajectory

Unlike many other 3D reconstruction tasks, where the main objective is to recover the shape of a static scene or object, we aim to reconstruct a drone trajectory which should be a continuous curve in 3D space. Thus we propose to exploit spline as an alternative representation of drone trajectory rather than a set of individual points.

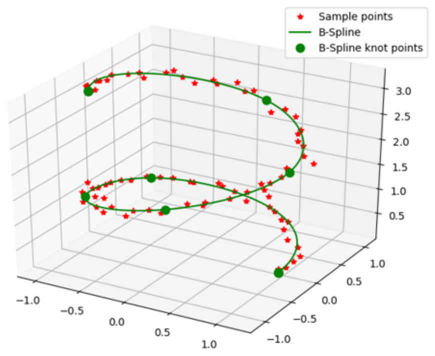


Figure 4.3: An example of B-spline in presence of noise

More specifically, we leverage cubic B-splines in this work which are commonly used for curve fitting. Given a set of 3D points in space, a B-spline can be fitted which is smooth and close to the given points with only a few control points. One main advantage of this spline representation is the resulting implicit geometric constraint. Without spline, drone detections in each frame are considered independent from each other and theoretically can be anywhere in space. The geometric constraint behind a spline follows the assumption that the sampled data should form a smooth continuous curve, which can be generally applied to drone trajectories. In this way, noises and local discontinuities could be filtered or reduced while the main structure of the trajectory remains unaffected.

Another important feature of B-spline is the local support. As a B-spline is defined as piecewise polynomial functions, change of curve in one location (e.g. via change of control point or coefficient) will not affect the whole curve. Thus any detection on a drone trajectory is dependent only on the local control points and coefficients. This property helps keep the sparsity character of Jacobian matrix in bundle adjustment. For the implementation in this work, a B-spline is fitted with a proper smooth factor using all these 3D points, when the initial trajectory is reconstructed or each time more points are triangulated. This smooth factor is manually selected such that the spline is defined by very few control points while it visually conforms closely to the 3D trajectory reconstructed as independent points. After coordinates of control points of a spline is optimized in bundle adjustment, the 3D points are obtained by resampling from the spline.

4.2.3 Spatiotemporal bundle adjustment

Bundle adjustment is an optimization step that simultaneously refines reconstructed 3D points, relative camera poses and camera optical parameters. In this work, we denote this process as spatiotemporal bundle adjustment as parameters

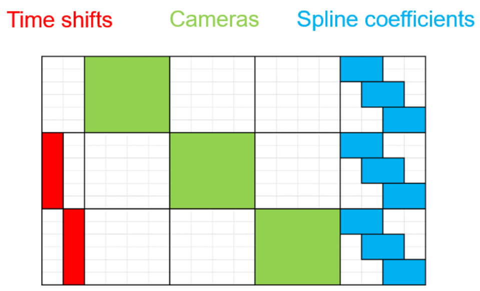


Figure 4.4: Sparse Jacobian matrix with spline coefficients

of camera synchronization will also be optimized. Thus the standard optimization minimizing reprojection error is modified as follows

$$\arg\min_{P,X,\beta} \sum_{t=1}^{T} \sum_{c=1}^{C} V_c^t \left\| \pi(P_c, X(t)) - x_c(t_c + \beta_c) \right\|^2 \tag{4.12}$$

where P_c denotes camera parameters, $X(t)$ the 3D location of drone at time t, V_c^t the binary indicator of point-camera visibility. t_c is the local time of camera c at which the drone is projected and β_c is the time shift of camera c w.r.t. the global timeline, such that $t = t_c + \beta_c$. Thus $x_c(t_c)$ is the original detection and $x_c(t_c + \beta_c)$ is the actual projection of the drone at time t. This optimization is referred to as S_{Points} as the drone trajectory is considered as a set of independent 3D points.

In section 4.2.2 a spline representation for drone trajectory is introduced. In this case, the cost function of the spatiotemporal bundle adjustment is revised as

$$\arg\min_{P,S,\beta} \sum_{t=1}^{T} \sum_{c=1}^{C} V_c^t \left\| \pi(P_c, S(k_t)) - x_c(t_c + \beta_c) \right\|^2 \tag{4.13}$$

where $S(k_t)$ denotes the 3D point sampled from the spline S at the global time t, which is affected by spline coefficients k_t. This equation shows that only the spline coefficients will be optimized, while the knot vector of the spline remains unchanged. Because otherwise the number of spline parameters is not fixed, which increases the complexity of the optimization. As mentioned in section 4.2.2 the Jacobian matrix of the spatiotemporal bundle adjustment with spline will be kept as a sparse matrix, as the reprojection error of one single point will not be determined by the entire spline coefficients (see Fig 4.4).

5 Experiments

This chapter presents various experiments of methods implemented in this work. It is divided into two sections. The first section evaluates the performance of camera synchronization on a synthetic dataset. The second section presents the final results of drone trajectory reconstruction on real datasets.

5.1 Camera Synchronization on Synthetic Data

In this section, experiments of methods for camera synchronization in section 4.1 are presented. These experiments are based on a synthetic dataset, as the accessible real datasets do not have the ground truth, i.e. not strictly hardware synchronized.

5.1.1 Synthetic Data

Although theoretically a synthetic trajectory can be easily generated randomly, we exploit a real drone trajectory that is reconstructed from two camera views. This trajectory is then projected onto the two known cameras resulting in two synchronized image detections. By doing so, this drone trajectory does not have strong violations of the underlying physical rules and any computed time shifts have a real physical reference (i.e. unit second) rather than only in numbers of frames. Fig 5.1 shows the synthetic 3D trajectory and Fig 5.2 shows the detections

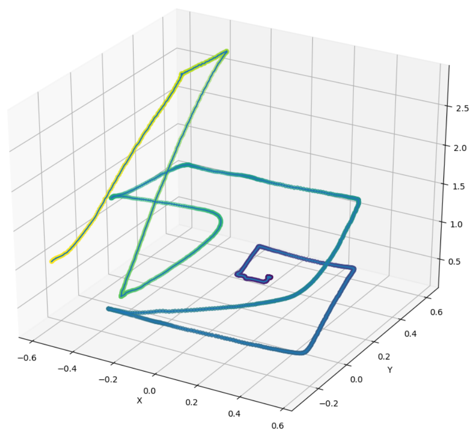
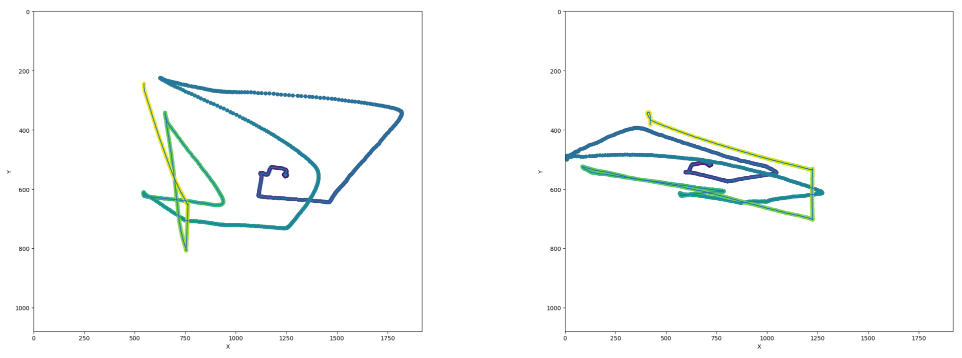


Figure 5.1: Synthetic trajectory

of this trajectory in two known cameras. Both cameras have the same frame rates of 30fps.



(a) Left camera (b) Right camera

Figure 5.2: 2D detections of the synthetic drone trajectory from two cameras

5.1.2 Performance of the quasi-minimal solver

The performance of the quasi-minimal solver originated from Albl et al. (2017) is investigated on this synthetic data. The two image detections are shifted towards each other with different numbers of frames resulting in ground truth time shift β_{gt}, where $\beta_{gt} \in [-50, 50]$ frames. We tested the solver with different interpolation distances d in powers of 2 and compared them also with the standard 8-point algorithm without synchronization. Image noises are added to both detections from a normal distribution with $\sigma = 1$ pixel. For each ground truth β_{gt} and interpolation distance d, the solver is tested 20 times and the average results are presented.

Fig 5.3 shows the results of our experiments. In general the solver performs well in terms of estimating the time shift β. The interpolation distance d has a strong impact on the range in which β can be correctly estimated. For a given d, the solver can estimate correct β at least up to d. For β_{gt} larger than d, the estimated time shifts are smaller than β_{gt}, but not higher. This suggests that d can be safely increased for better estimates.

The ratio of inliers is a good indicator of whether β_{gt} is correctly estimated. For $d = 1$ and $d = 2$, even though the ratio of inliers decreases with large shifts, β can still be correctly estimated, up to about 10 frames. From Fig 5.3 we can see that there are two peaks with respect to ratio of inliers, one at $\beta_{gt} = 0$ and the other at $\beta_{gt} = d$. This makes sense as the tangent vector v will pass through this

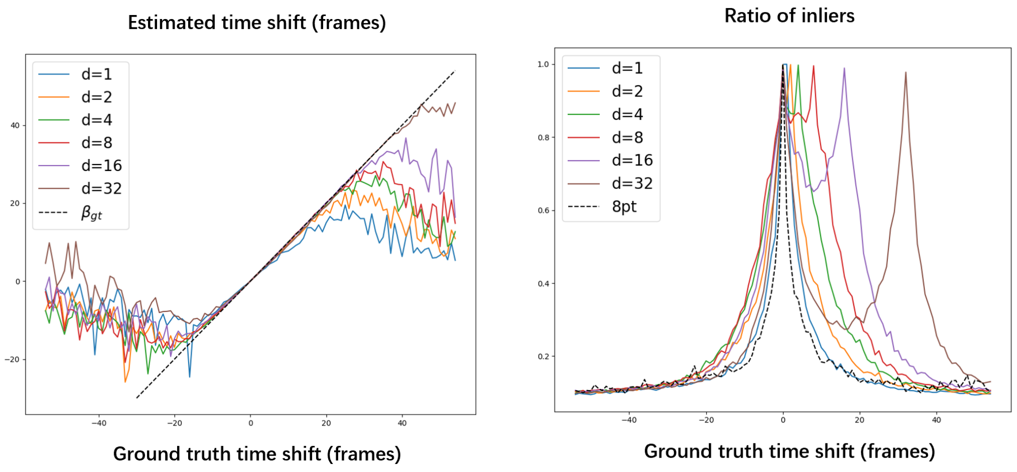


Figure 5.3: Performance of the quasi-minimal solver in terms of ratio of inliers (left) and estimated time shift (right). Results with various ground truth β_{gt} and interpolation distance d, averaged from 20 runs.

next dth point. Another issue worth to be mentioned is the inferior performance when $\beta_{gt} < 0$. This means the estimation of β would fail when d is in the wrong direction w.r.t β_{gt}, which can be solved by searching over d in both directions.

5.1.3 Performance of iterative algorithm and fixed-time algorithm

The iterative and the fixed-time algorithm are also experimented on this synthetic dataset. As both algorithms are intended to estimate large time shifts, the extent of ground truth time shifts is set up to $\beta_{gt} \in [-200, 200]$ frames, which corresponds to $\beta_{gt} \in [-6.7, 6.7]$ seconds. For the iterative algorithm, the maximal exponent of powers of 2 for the interpolation distance is set to 8 such that $d \in \{\pm 2^0, \pm 2^1, \cdots, \pm 2^8\}$. For the first stage of the fixed-time algorithm, the step length for d is set to 20 frames and the resulting search vector is $d \in \{-200, -180, \cdots, -1, 1, \cdots, 180, 200\}$. For the second stage $d = \pm 1$ is applied. For each ground truth time shift β_{gt} we ran both algorithms 10 times and recorded the rates with which the two cameras are successfully synchronized up to single frame accuracy. Results can be seen in Fig 5.4. It is apparent that the fixed-time algorithm is very robust for β_{gt} in a range of around -4 to 4 seconds, but starts to be unstable rapidly for larger time shifts. On the other hand, the iterative algorithm generally outperforms the fixed-time algorithm for these larger time shifts, but can not ensure correct estimates at each run time. So a feasible strategy is applying the fixed-time algorithm when two cameras are closely pre-synchronized and the remainder β is relatively small. One can benefit from the iterative algo-

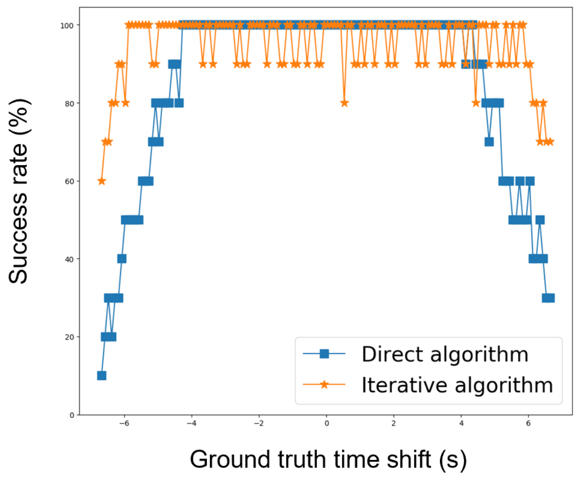


Figure 5.4: Success rates of the iterative algorithm and the fixed-time algorithm. For each ground truth time shift β_{gt} the rates are presented from 10 runs with which β_{gt} is correctly estimated up to one frame.

rithm when the manual pre-synchronization is not accurate and the remainder β still has a couple of seconds.

5.2 Trajectory Reconstruction on Real Data

In this section, experiments of our trajectory reconstruction pipeline, which are conducted in different configurations on real datasets, are presented. The ground truth positions of the real drone trajectories are available, which enables quantitative comparisons and statistical analysis.

5.2.1 Real Data

Our real datasets contain two drone flights both in a wild outdoor environment. Durations of both flights, in which the drone is captured by cameras, are around 2 minutes. The Real-time Kinematic (RTK) positioning data of both flights are available, provided by the company *Fixposition* [1]. The raw RTK data are in form of geodetic datum, i.e. longitude, latitude and height. These are then transformed into local ENU (East, North, up) coordinates with an arbitrarily selected local reference point. These RTK data are regarded as ground truth in the rest of this work, as they have very high accuracy in centimeter level, based on information on the official website of Fixposition (see Table 5.1).

Our multi-view system contains four static cameras on the ground. Fig 5.5 shows

[1]Official website: www.fixposition.ch

Horizontal accuracy	Vertical accuracy	Measurements rate
1cm + ppm	1.5cm + ppm	5Hz

Table 5.1: Specifications of RTK data according to Fixposition

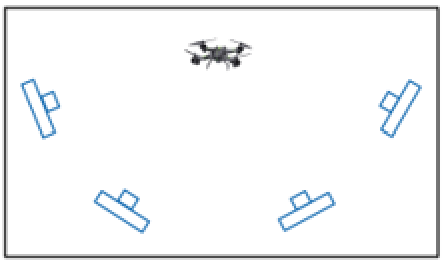


Figure 5.5: Camera configuration roughly displayed from vertical view

a rough configuration of the cameras, which is valid for both flights. The distance of each camera to the center of drone flight is less than 100m. Although our four cameras captured the drone with different frame rates, these frame rates are known (30, 30, 50, 25) and the resulting 2D detections from each camera are interpolated such that they correspond to a frame rate of 30fps. On the other hand, cameras are triggered at different times on purpose and the time shifts between them are unknown, i.e. no ground truth for camera synchronization.

As object detection from videos is not a part of this work, the 2D drone detections from each camera are provided as input data (see Fig 5.6). The number of detections from each camera varies roughly from 2000 to 4000 points depending on how long the drone flew inside the field of view of each camera. The appearance of the drone has different scales due to different distances to each camera and it makes detection and tracking more difficult. Therefore noise can be expected from input detections, which is a good challenging situation for our reconstruction.

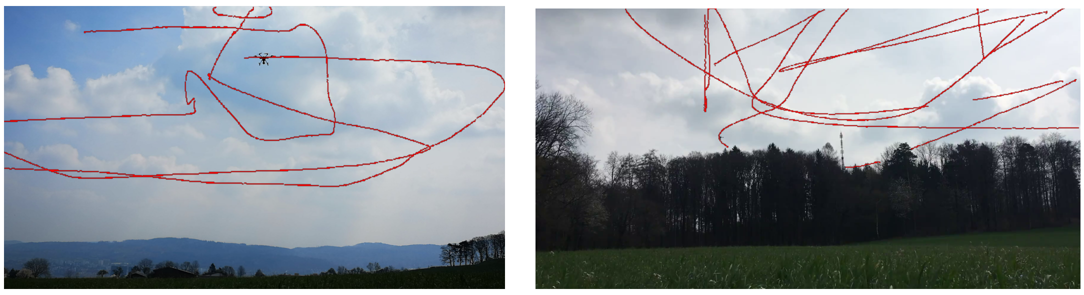


Figure 5.6: Example of input drone detections

5.2.2 Experimental configuration

As two drone flights are available, we denote them simply as 1st Flight and 2nd Flight. For both flights, the input drone detections are roughly pre-synchronized by means of manual alignment, such that relative time shift between each pair is less than 10s. Because manual synchronization of cameras in a range of 10s is feasible, either based on visual or acoustic contents of videos. And a time shift large than 10s is considered out of the capability of our algorithms. After cameras are globally synchronized either with the iterative or the fixed-time algorithm, we started the incremental reconstruction with two views that have most corresponding detections and sequentially add another view that has most correspondences to the current triangulated trajectory. For the bundle adjustment, we used the *least_squares* function from the Scipy Python library with maximum number of 20 iterations.

As a spline representation for drone trajectory is proposed in section 4.2.2, this is experimented together with the standard approach. We denote reconstruction based on spline as *Spline* and reconstruction as a set of independent 3D points as *Points*. Our reconstructed drone trajectories are eventually transformed through a similarity transformation, which enables a comparison with the ground truth RTK data in the ENU coordinate system. As the measurement rate of our reconstruction (30fps) is higher than that of the RTK data (5Hz), different alignments between both data sources are possible, i.e. one RTK measurement has six possible corresponding points from the reconstructed trajectory. Thus we estimate all possible transformations and choose the one that gives the smallest mean error of drone locations.

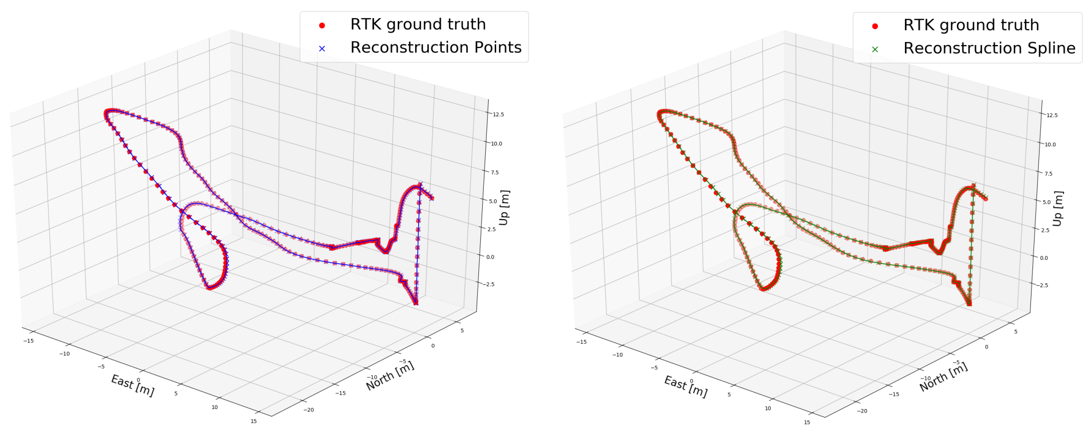
In this work, the four cameras capturing the drone are calibrated using the standard method with a chessboard (Zhang, 2000). Additionally we also did reconstruction from partial-calibrated cameras, i.e. assuming square pixels, principle points in image center and no radial distortions. Theoretically focal lengths of cameras can be estimated by the Eq. 3.11, yet this has been proved as very sensitive and not robust. Thus we applied approximated values for focal lengths that deviate 20% from the calibrated values as a simulation of uncalibrated cameras.

5.2.3 Qualitative and quantitative results

In this section, results of camera synchronization and drone trajectory reconstruction are presented for 1st Flight and 2nd Flight separately. Also results of reconstruction using uncalibrated cameras are presented.

	Cam1	Cam2	Cam3	Cam4
Manual	0	46	-8	109
Manual (optimized)	0	46.53	-11.27	110.93
Iterative	0	33.63	-18.25	121.89
Iterative (optimized)	0	33.58	-17.93	122.06
Fixed-time	0	32.55	-17.68	124.12
Fixed-time (optimized)	0	33.26	-17.56	122.54

Table 5.2: Camera synchronization of 1st Flight. All time shifts are in unit frame and relative to the first camera. Both initial and optimized estimates are listed for all methods. The iterative algorithm and the fixed-time algorithm give very close estimates of time shifts, while manual synchronization differs significantly.



(a) Points (b) Spline

Figure 5.7: Reconstruction of 1st Flight compared with RTK ground truth without (left) and with (right) spline representation.

Results of 1st Flight The relative time offsets (frames) between cameras are presented in Table 5.2. Raw estimates from all methods as well as optimized estimates after bundle adjustment (without spline representation) are provided. The starting time of the first camera is regarded as the reference for all computed time shifts. Manual synchronization of two cameras is based on visual contents and acoustic information from videos, which is not precise enough and does not converge to actual time shifts after optimization. On the other hand, it is apparent that both the iterative and the fixed-time algorithm give very close estimates of time shifts, with differences less than about 1 frame. As there is no ground truth for these time shifts, we compared the raw estimates with those after optimization. The changes of time shifts caused by optimization are small, i.e. less than 1.5 frames. This could infer that the initial estimates are close enough to the optimal values, but it is not a certain conclusion. To confirm the correctness of our results,

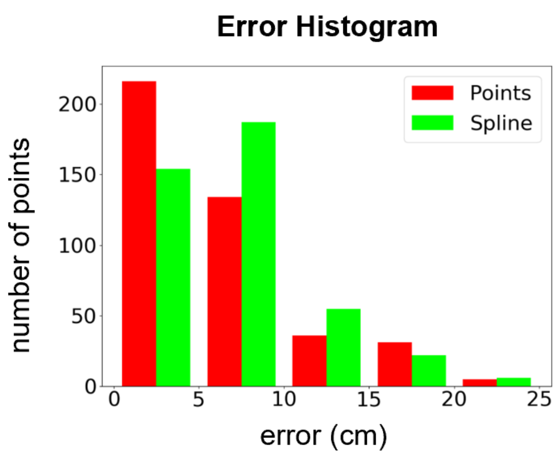


Figure 5.8: Error histogram of the 1st Flight

	Mean (cm)	RMSE (cm)	Max (cm)	Min (cm)
Points	6.7	5.0	26.2	0.5
Spline	7.3	4.6	26.0	0.5

Table 5.3: Mean, RMSE and max errors of the 1st Flight reconstruction

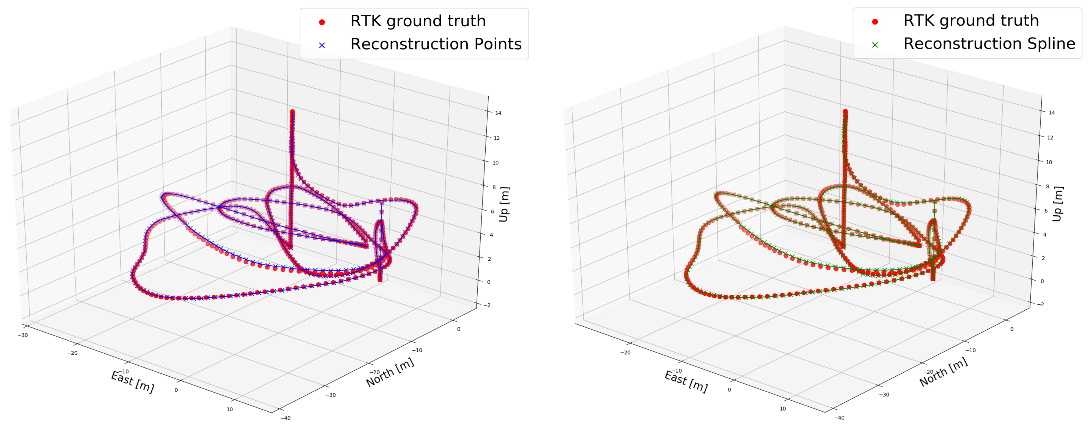
we will present the drone reconstructions following camera synchronization with the iterative algorithm.

Fig. 5.7 shows qualitative results of reconstruction of the 1st Flight. We can see that our reconstructed trajectory conforms well to the ground truth such that they nearly overlap each other. This is valid for both reconstruction methods without and with spline representation. Table 5.3 and Fig 5.8 present statistic analysis of the 1st Flight. The majority of the reconstructed points has error less than 10cm. Although the mean position error of reconstruction using spline is larger than that of normal reconstruction, the RMSE and the maximum error are lowered with spline representation.

Results of 2nd Flight The results of the 2nd Flight are presented in an analogous manner as the 1st Flight. Table 5.4 shows the estimated time shifts between the cameras. Though carried out carefully, manual synchronization still has inevitable distinct deviations from the actual time shifts. Differences of the initial estimates provided by the iterative and the fixed-time algorithm are smaller than 1 frame, which is an expected behavior similar to the 1st Flight. Yet the initial estimates are changed around 2 frames or more after optimization (without spline representation). One possible reason could be the noise in the input detections. As the time shift optimization of each camera is affected by all the reprojection errors from that camera, the modification of time shift could act as a compensation to

	Cam1	Cam2	Cam3	Cam4
Manual	0	-54	30	97
Manual (optimized)	0	-52.16	27.49	102.28
Iterative	0	-71.54	14.29	105.42
Iterative (optimized)	0	-73.66	12.47	104.36
Fixed-time	0	-71.48	14.26	106.12
Fixed-time (optimized)	0	-73.97	12.39	104.31

Table 5.4: Camera synchronization of 2nd Flight. All time shifts are in unit frame and relative to the first camera. Both initial and optimized estimates are listed for both all methods. Similar to the 1st Flight, manual alignment shows relatively large deviations from the actual time shifts, wherears both the iterative algorithm and the fixed-time algorithm provide similar initial estimates of time shifts. Bundle adjustment leads to changes of time shift estimates around 2 frames.



(a) Points (b) Spline

Figure 5.9: Reconstruction of 2nd Flight compared with RTK ground truth without (left) and with (right) spline representation.

noise in other parameters to reduce the total reprojection error.

Fig. 5.9 shows qualitative results of reconstruction of the 2nd Flight. Once again our reconstructed trajectories coincide with the ground truth barely with distinct margins. The trajectory optimized as a spline appears very similar to the trajectory optimized as a set of independent points. Table 5.5 and Fig. 5.10 present quantitative results of the 2nd Flight. The majority of the reconstructed points has error between 5 to 15cm, which is slightly larger than the 1st Flight. This time the spline representation benefits the trajectory reconstruction in all three criteria of mean, RMSE and maximum errors. Especially the spline representation contributes to a reduction of the maximum error about 6cm. Visualizations of position errors along the trajectory for both flights can be found in Appendix A

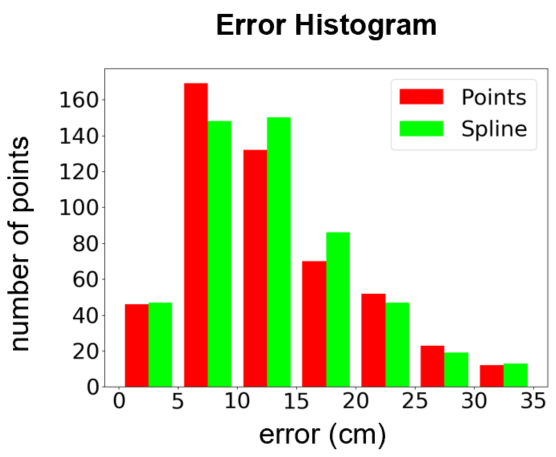


Figure 5.10: Error histogram of the 2nd Flight

	Mean (cm)	RMSE (cm)	Max (cm)	Min (cm)
Points	13.2	7.4	40.0	1.6
Spline	12.9	6.7	34.4	1.4

Table 5.5: Mean, RMSE and max errors of the 2nd Flight reconstruction

Results of uncalibrated cameras As mentioned in section 5.2.2, we also experimented trajectory reconstructions using uncalibrated cameras with several prior assumptions. Table 5.6 shows the results in form of the mean position errors in centimeters. It is expected that the reconstructions with uncalibrated cameras are inferior, which means larger errors in tens of centimeters. Trajectories optimized as spline present slightly better reconstructions with smaller errors by around 3cm.

	1st Flight (points)	1st Flight (spline)	2nd Flight (points)	2nd Flight (spline)
Calibrated	6.7	7.3	13.2	12.9
Uncalibrated	25.2	22.4	79.1	76.8

Table 5.6: Error of reconstruction with uncalibrated cameras. The mean position errors (cm) to the ground truth RTK data in centimeters for both flights are reported.

6 Discussion

In this work, the first main task is temporal synchronization between two cameras. The implemented quasi-minimal solver is an extension of the popular Eight-point algorithm. The estimation of time shift β is based on the generalized eigenvalue solution. Theoretically, once the time shift β is estimated, the sequences of input drone detections can be aligned and thus the fundamental matrix can be estimated separately. So the time shift and the fundamental matrix can be computed simultaneously, but not necessarily. One advantage of this effect is the possibility of using different thresholds for β and the fundamental matrix. For the fixed-time algorithm for large time shifts, results have shown that its performance declines rapidly for large interpolation distances d. One possible reason is that the effective ranges for different d are not identical. So the search interval for d is not necessarily uniform. For large time shifts, a denser search for d can potentially give better estimates, yet will clearly increase the computational burden. For the iterative algorithm, the maximal exponent of powers of 2 applied to d affects the computational cost of this algorithm in a great part. So it should be chosen carefully, e,g. according to the initial guess of β, not just as large as possible. During experiments, two cameras capturing real drone trajectories have frame rates very close to 30fps according to display on hardware, e.g. 29.94fps. We have found that this deviation is not negligible because the error in time scale will accumulate through time potentially resulting in large time differences afterward. Future work of estimating the relative time scale is desired.

A contribution of this work is the spline representation of drone trajectory. The spline smoothing acts as a regularizer that implicitly enforces geometric constraints to individual samples from trajectories. The results of our experiments indicate that the spline representation mitigates noise and helps reduce large errors. In most experiments, trajectories optimized as spline have smaller RMSE and maximum errors. Yet for the 1st Flight the trajectory optimized as spline has a slightly larger mean position error compared with the standard reconstruction without spline. One possible reason is that the standard method can already recover the trajectory very closely given good initial drone detections. As there is a trade-off between smoothness and closeness when creating a spline, using spline might cause larger errors if given samples are highly accurate and/or the smoothing factor is

	Noise $\sigma = 0$	Noise $\sigma = 3$
Points	6.7	10.4
Spline	7.3	9.6

Table 6.1: Mean position error (cm) of the 1st Flight with noise. A noise with $\sigma = 3$ is added to raw detections and the averaged results of 5 runs of our method are listed

too large to preserve local geometric features. Another source of position errors is the imperfection of the RTK data. As our reconstructions reach a high accuracy of several centimeters, the placement of the RTK onboard sensor should be taken into account. Because the center point of the drone is tracked in our 2D detections, which might not coincide with the location of the RTK sensor and thus leads to errors in centimeter level.

In this work, only coordinates of control points are optimized as spline parameters. The number of control points is fixed and is determined by the initial smoothing factor. So one can set up the initial spline by tuning the smoothing factor for a balance between smoothness and closeness. The number of control points will be increased until the smoothing condition is satisfied:

$$\sum \|y_i - spline(x_i)\|^2 < s \tag{6.1}$$

where s is the smoothing factor, x_i are input points and y_i are approximated points from the spline at the same frames as x_i. In our experiments, we chose the smoothing factor s such that the spline visually coincides well to the original points while control points are much less than the original points, i.e. more than 3000 points can be represented by a spline with less than 100 control points. As this s is empirically tuned in this work, the impact of s deserves a detailed statistic analysis in the future.

In section 5.2.3 trajectories are reconstructed from raw image detections. To test the robustness of our reconstruction pipeline, especially the spline representation, experiments should have been done where noise is added to input detections. Unfortunately, a thorough analysis with different levels of noise has not been accomplished in this work due to the limitation of time and workload for this thesis. We tested a configuration for the 1st Flight where image noise is added to detections from a normal distribution with $\sigma = 3$ pixels and ran it 5 times (see Table 6.1). The trajectory optimized as a spline has a mean position error of 9.6cm and is closer to the ground truth RTK data than the normal reconstruction, which has a mean error of 10.4cm. Due to the reconstructed trajectories, the extent of camera distances to the drone is between 10m to 60m. For the entire setup on this scale, our experiments show that our reconstruction pipeline, especially with spline representation, does not require input detections to be extremely accurate. Yet as mentioned above, further experiments w.r.t robustness are anticipated.

Another interesting aspect of the reconstruction is the better result of the 1st Flight compared to the 2nd Flight. One possible factor for that is visibility. A point from the 1st Flight ist captured by 3.1 cameras on average, whereas this number is 2.8 for the 2nd Flight. A trajectory can be better recovered if it is recorded by more cameras, which provides useful redundancies. Other relevant factors include the physical motion of the drone, distances of the drone to the cameras, etc. Further experiments could be made for in-depth investigation.

7 Conclusion & Outlook

The main objective of this thesis is to explore the potential of reconstructing drone trajectories observed from multiple stationary cameras. Sequences of drone detections from each camera are provided as the only input. A Euclidean reconstruction is computed and compared with the ground truth trajectory from RTK data. We have shown that our reconstructions can achieve high accuracy in tens of centimeters.

The first contribution of this work involves camera synchronization. A solver from Albl et al. (2017) is implemented that can simultaneously estimate the time shifts between two cameras and their relative camera pose. The solver relies purely on image correspondences and is robust to noise and outliers exploiting RANSAC. Based on this solver, two algorithms are implemented to estimate large time offsets (tens to hundreds of frames, up to about 7s). The iterative method is capable of synchronizing two sequences step by step and is thus suitable for large time shifts. The fixed-time algorithm is more efficient and robust for estimating time shifts in a certain range. Both algorithms are experimented on a synthetic dataset and show high success rates to synchronize two cameras with single frame accuracy.

The camera synchronization is regarded as a pre-processing step of our reconstruction pipeline. We followed the incremental principle of multi-view reconstruction and proposed a spline representation of drone trajectory for bundle adjustment. The underlying idea is to consider a drone trajectory as a smooth curve rather than independent points, which can be well approximated by a spline with few parameters. We compared our spline representation with the standard reconstruction on real datasets with ground truth. Experiments have indicated that a smoothing spline contributes to mitigating the variance of position errors and reduction of extreme errors, while the efficiency of sparse bundle adjustment is not affected much. We showed that our reconstructions, either with or without spline representation, conform to the RTK ground truth with deviations within tens of centimeters. Reconstructions from uncalibrated cameras have also been proved possible albeit with inferior results.

In general, this work is an attempt with decent results towards the promising direction of drone trajectory reconstruction with vision-based approaches. Further investigations should be conducted to refine or extend the methods from this thesis. For instance, a method of camera synchronization allowing estimation of relative time scale could improve the temporal alignment of drone detections. Because even the displayed frame rates from hardware are only precise up to few decimal places. To improve the robustness of reconstructions, experiments with different amounts of noise for input detections, camera poses or camera synchronizations could be performed. An important aspect is studying the behavior of reconstruction accuracies when drones fly in different spatial scales and with various distances to cameras. For better applicabilities, the potential of uncalibrated cameras could be further explored. In the best scenario, reconstruction could be possibly accomplished in real-time or online applications.

Reference

Agarwal, S., Snavely, N., Seitz, S. M., and Szeliski, R. (2010). "Bundle adjustment in the large". In: *European conference on computer vision*. Springer, pp. 29–42.

Albl, C., Kukelova, Z., Fitzgibbon, A., Heller, J., Smid, M., and Pajdla, T. (2017). "On the two-view geometry of unsynchronized cameras". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4847–4856.

Avidan, S. and Shashua, A. (2000). "Trajectory triangulation: 3D reconstruction of moving points from a monocular image sequence". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.4, pp. 348–357.

Bougnoux, S. (1998). "From projective to euclidean space under any practical situation, a criticism of self-calibration". In: *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, pp. 790–796.

Brand, M., Antone, M., and Teller, S. (2004). "Spectral solution of large-scale extrinsic camera calibration as a graph embedding problem". In: *European Conference on Computer Vision*. Springer, pp. 262–273.

Brown, D. C. (1966). "Decentering distortion of lenses". In: *Photogrammetric Engineering and Remote Sensing*.

Caspi, Y., Simakov, D., and Irani, M. (2006). "Feature-based sequence-to-sequence matching". In: *International Journal of Computer Vision* 68.1, pp. 53–64.

Caspi, Y. and Irani, M. (2002). "Spatio-temporal alignment of sequences". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.11, pp. 1409–1424.

Chum, O., Matas, J., and Kittler, J. (2003). "Locally optimized RANSAC". In: *Joint Pattern Recognition Symposium*. Springer, pp. 236–243.

Cohen, A., Zach, C., Sinha, S. N., and Pollefeys, M. (2012). "Discovering and exploiting 3d symmetries in structure from motion". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 1514–1521.

Cui, H., Gao, X., Shen, S., and Hu, Z. (2017). "HSfM: Hybrid structure-from-motion". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1212–1221.

Dai, C., Zheng, Y., and Li, X. (2006). “Subframe video synchronization via 3d phase correlation”. In: *2006 International Conference on Image Processing*. IEEE, pp. 501–504.

Fukunaga, K. and Hostetler, L. (1975). “The estimation of the gradient of a density function, with applications in pattern recognition”. In: *IEEE Transactions on information theory* 21.1, pp. 32–40.

Gong, Y., Meng, D., and Seibel, E. J. (2015). “Bound constrained bundle adjustment for reliable 3D reconstruction”. In: *Optics express* 23.8, pp. 10771–10785.

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision.* Cambridge university press.

Hartley, R., Silpa-Anan, C., et al. (2002). “Reconstruction from two views using approximate calibration”. In: *Proc. 5th Asian Conf. Comput. Vision.* Vol. 1, pp. 338–343.

Hartley, R., Gupta, R., and Chang, T. (1992). “Stereo from uncalibrated cameras”. In: *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* IEEE, pp. 761–764.

Hartley, R. I. (1995). “In defence of the 8-point algorithm”. In: *Proceedings of IEEE international conference on computer vision.* IEEE, pp. 1064–1070.

Hartley, R. I. and Sturm, P. (1997). “Triangulation”. In: *Computer vision and image understanding* 68.2, pp. 146–157.

Lepetit, V., Moreno-Noguer, F., and Fua, P. (2009). “Epnp: An accurate o (n) solution to the pnp problem”. In: *International journal of computer vision* 81.2, p. 155.

Lourakis, M. I. and Argyros, A. A. (2009). “SBA: A software package for generic sparse bundle adjustment”. In: *ACM Transactions on Mathematical Software (TOMS)* 36.1, p. 2.

Marquardt, D. W. (1963). “An algorithm for least-squares estimation of nonlinear parameters”. In: *Journal of the society for Industrial and Applied Mathematics* 11.2, pp. 431–441.

Nischt, M. and Swaminathan, R. (2009). “Self-calibration of asynchronized camera networks”. In: *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops.* IEEE, pp. 2164–2171.

Nistér, D. (2004). “An efficient solution to the five-point relative pose problem”. In: *IEEE transactions on pattern analysis and machine intelligence* 26.6, pp. 0756–777.

Noguchi, M. and Kato, T. (2007). "Geometric and timing calibration for unsynchronized cameras using trajectories of a moving marker". In: *2007 IEEE Workshop on Applications of Computer Vision (WACV'07)*. IEEE, pp. 20–20.

Padua, F., Carceroni, R., Santos, G., and Kutulakos, K. (2008). "Linear sequence-to-sequence alignment". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.2, pp. 304–320.

Rozantsev, A., Sinha, S. N., Dey, D., and Fua, P. (2017). "Flight dynamics-based recovery of a uav trajectory using ground cameras". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6030–6039.

Sinha, S. N. and Pollefeys, M. (2010). "Camera network calibration and synchronization from silhouettes in archived video". In: *International journal of computer vision* 87.3, pp. 266–283.

Su, Z. *Fixposition*. CEO & Co-founder. `https://www.fixposition.ch/`. Accessed: 2019-06-30.

Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (1999). "Bundle adjustment—a modern synthesis". In: *International workshop on vision algorithms*. Springer, pp. 298–372.

Valmadre, J. and Lucey, S. (2012). "General trajectory prior for non-rigid reconstruction". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 1394–1401.

Vo, M., Narasimhan, S. G., and Sheikh, Y. (2016). "Spatiotemporal bundle adjustment for dynamic 3d reconstruction". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1710–1718.

Wu, C. (2013). "Towards linear-time incremental structure from motion". In: *2013 International Conference on 3D Vision-3DV 2013*. IEEE, pp. 127–134.

Yan, J. and Pollefeys, M. (2004). "Video synchronization via space-time interest point distribution". In: *Advanced Concepts for Intelligent Vision Systems*. Vol. 1, pp. 12–21.

Yuan, C. and Medioni, G. (2006). "3D reconstruction of background and objects moving on ground plane viewed from a moving camera". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. IEEE, pp. 2261–2268.

Zhang, Z. (2000). "A flexible new technique for camera calibration". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.

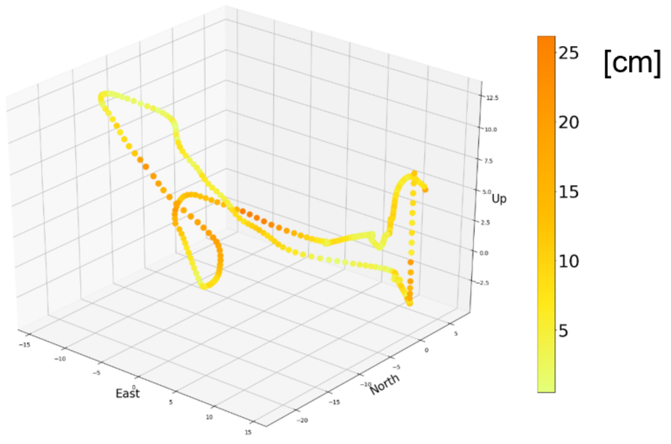
Zheng, E., Ji, D., Dunn, E., and Frahm, J.-M. (2015). “Sparse dynamic 3d reconstruction from unsynchronized videos”. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4435–4443.

Zhu, Y. and Lucey, S. (2013). “Convolutional sparse coding for trajectory reconstruction”. In: *IEEE transactions on pattern analysis and machine intelligence* 37.3, pp. 529–540.

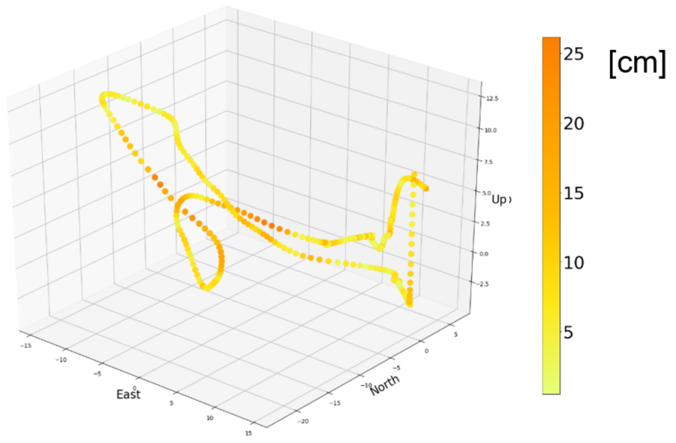
Zivkovic, Z. and Van Der Heijden, F. (2006). “Efficient adaptive density estimation per image pixel for the task of background subtraction”. In: *Pattern recognition letters* 27.7, pp. 773–780.

Appendix A

Reconstruction errors along trajectory

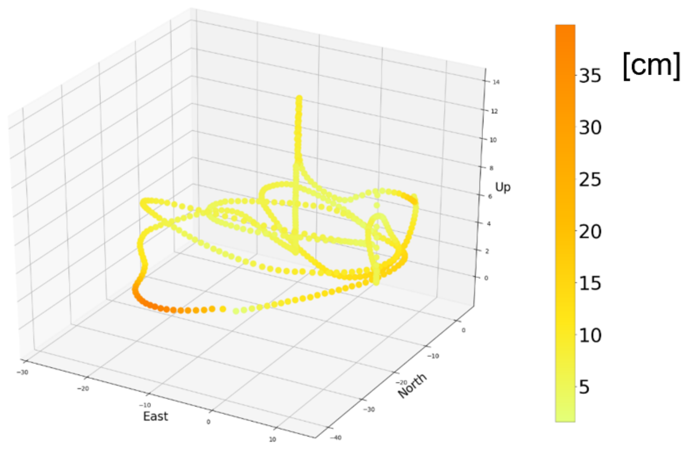


(a) Points

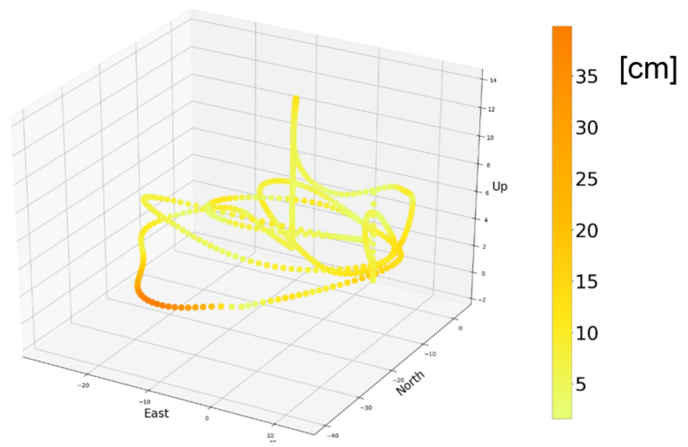


(b) Spline

Figure A.1: Reconstruction error of 1st Flight along the trajectory compared with RTK ground truth without (top) and with (down) spline representation.



(a) Points



(b) Spline

Figure A.2: Reconstruction error of 2nd Flight along the trajectory compared with RTK ground truth without (top) and with (down) spline representation.