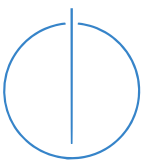
DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

Improving Localization of a Multicopter by External Tracking

Simon Boche



DEPARTMENT OF INFORMATICS

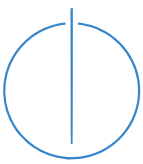
TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

Improving Localization of a Multicopter by External Tracking

Verbesserte Lokalisierung eines Multikopters durch Externes Tracking

Author:	Simon Boche
Supervisor:	PD Dr. habil. Rudolph Triebel
Advisors:	Dr. Wolfgang Stürzl,
	Dipl.-Ing. Florian Steidle
Submission Date:	15 Dec 2020



Declaration

I confirm that this master's thesis with the title *Improving Localization of a Multicopter by External Tracking* is my own work and I have documented all sources and material used.

Simon Boche, Munich, December 15, 2020

Abstract

Collaborative teams of heterogeneous robots will be a key technology for future planetary exploration missions. Therefore, a high degree of local autonomy is required. In this work, a concrete use case involving a Rover System (LRU) and a Micro Aerial Vehicle (ARDEA) is presented. It is assumed that during an exploration flight of the flying robot, an interesting spot is detected which is supposed to be further investigated by scientific tools of the ground robot. For that purpose, the relative orientation and translation between the two systems need to be determined with a high accuracy by the use of on-board sensor readings. The task of pose estimation can be formulated as a modified and temporal version of a Perspective-n-Point (PnP) problem. While in a classic PnP formulation, the pose can be estimated from known 3D-2D correspondences in one image frame, the proposed approach formulates the problem in terms of 3D-2D correspondences belonging to a flight trajectory over time.
An essential part of the overall problem is given by detecting and tracking ARDEA in a sequence of images to retrieve 2D observations. To address this issue, two different approaches are proposed. One uses a combination of Background Subtraction and Correlation Filter based tracking (CFTs). The other approach is based on the RetinaNet detector which is trained on the task of detecting ARDEA in an image. Both approaches are evaluated on experimental data to assess and compare their performances.
For the task of pose estimation, an existing approach aware of 2D uncertainties in the image plane is extended to also incorporate 3D uncertainties originating from VO measurements. The pose of ARDEA in the camera frame of the LRU is computed by minimizing weighted residuals in a reduced observation space which is spanned by tangent vectors to the unit sphere in the camera frame. As a reliability measure, covariance estimates of the resulting pose parameteres are derived. The proposed approach for fully uncertainty-aware pose estimation is validated in simulation and in a real-world experiment. Results indicate the large potential of the proposed approach by significantly increasing the accuracy of the estimated poses and by yielding meaningful covariance estimates.

This thesis will introduce a framework for estimating the pose of ARDEA with respect to LRU based on image streams of cameras mounted on LRU and Visual Odometry readings from ARDEA.

Acknowledgements

I have had the unique opportunity to write my thesis within the Institute of Robotics and Mechatronics of the German Aerospace Center (DLR) in Oberpfaffenhofen. I want to thank all the smart and enthusiastic people at DLR that contributed to my work and enriched my studies by giving me the opportunity to gain insight into fascinating research projects.

First of all, I would like to thank my advisors, Dr. Wolfgang Stürzl and Florian Steidle, for the possibility to work on this topic, for providing me feedback and ideas in insightful technical discussions and for the great support throughout this thesis.

I further want to thank Dr. habil. Rudolph Triebel for supervision and for the possibility to write my thesis in the Department of Perception and Cognition.

Last but not least, special thanks is due to my family and my girlfriend who always encouraged me and helped me to achieve my goals.

Simon Boche

Contents

1. Introduction **1**
- 1.1. Problem Statement . . . 3
- 1.2. Contributions and Outline . . . 4

2. Related Work **5**
- 2.1. Visual Object Tracking . . . 5
- 2.2. Perspective-n-Point Problem . . . 7

3. Tracking Approach **9**
- 3.1. Theoretical Background . . . 9
  - 3.1.1. Background Subtraction: Mixture of Gaussians (MOG) . . . 9
  - 3.1.2. CSRDCF: Discriminative Correlation Filter with Channel and Spatial Reliability . . . 11
  - 3.1.3. RetinaNet . . . 12
- 3.2. Applied tracking algorithms . . . 14
  - 3.2.1. Conventional Tracker based on MOG and CFT . . . 14
  - 3.2.2. Tracking-by-detection using a RetinaNet Detector . . . 16

4. Pose Estimation **19**
- 4.1. Problem Formulation . . . 19
  - 4.1.1. Integration of Odometry Readings . . . 20
  - 4.1.2. Application of the Camera Model . . . 21
  - 4.1.3. Covariance Propagation . . . 22
  - 4.1.4. Reduced Observation Space . . . 24
- 4.2. Pose Estimation as Minimization Problem . . . 25
  - 4.2.1. Derivation of Minimization Problem . . . 25
  - 4.2.2. Linear Least-Squares Solutions . . . 26
  - 4.2.3. Non-Linear Optimization . . . 28
- 4.3. Overall Algorithm: PnP with Uncertainties . . . 29

5. Experimental Validation **31**
- 5.1. Prerequisites . . . 31
  - 5.1.1. Experimental Data Sets . . . 31
  - 5.1.2. Training a RetinaNet Detector . . . 32
- 5.2. Validation of Tracking Approaches . . . 35
- 5.3. Validation of Pose Estimation Approach . . . 41
  - 5.3.1. Validation Metrics . . . 41
  - 5.3.2. Simulation . . . 41
  - 5.3.3. Real-World Experiment . . . 45

6. Conclusion **49**
- 6.1. Summary . . . 49
- 6.2. Future Work . . . 51

List of Figures **53**

Bibliography **55**

A. Derivations for Pose Estimation **I**
A.1. Detailed Covariance Propagation for Static Camera I
A.2. Construction of Matrices for Covariance Propagation from VO Results I
A.3. Computation of the Jacobian for Spherical Normalization II

1 Introduction

The application of space robotics has enabled scientific breakthroughs and is a key technology to fulfilling the human ambition to explore extraterrestrial bodies such as Moon or Mars. Planetary exploration missions aim to find answers to fundamental scientific questions as for example about the origin of Earth or the potential existence of extraterrestrial life. However, developments in the field of space robotics do not only address these fundamental research areas, but they can also contribute to other scientific disciplines including applications like medical technology (surgical robotics), search and rescue or agricultural robotics [1]. Although space robotic missions may sound futuristic, they have quite a rich history. The first robotic system that has been successfully operated on an extraterrestrial body dates back to 1967 when the *Surveyor 3 lander* launched to Moon including an on-board robotic sampler [2]. Lately, especially rover missions have gained particular attention with space rover missions like NASA's *Perseverance Rover* (launched 2020) or ESA's *ExoMars Rover* (launch scheduled for 2022).

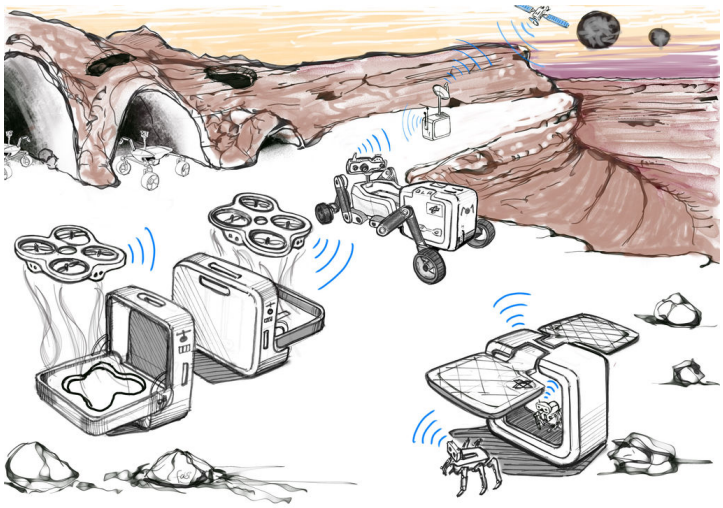


Figure 1.1.: Sketch of a planetary exploration mission using a heterogeneous team of robots [3].

Mainly two key attributes for space robotic systems can be identified [2]: locomotion and autonomy. Especially a high level of autonomy is essential as long round-trip communication times make teleoperation by humans difficult. These communication delays can range from several seconds (Moon) to several minutes (Mars) [4]. The required level of local autonomy can be supported by building heterogeneous teams of robots.
The ARCHES (*Autonomous Robotic Networks to Help Modern Societies*) project by the Helmholtz Association addresses this challenge. In cooperation with the Helmholtz Institutes AWI, GEOMAR and KIT, the DLR develops a heterogeneous team consisting of flying and

driving robots together with mobile infrastructure elements that can explore environments that are out of reach for humans. The sketch in Figure 1.1 visualises a possible scenario of a planetary exploration by a heterogeneous team of robots. These teams can benefit from complementary capabilities of every individual agent and can operate collaboratively to perform complex tasks in unknown and challenging surroundings [5]. There is a larger number of further benefits such as increased efficiency and robustness by parallelization and redundancy.

As part of the ARCHES project, a team of two driving robots and one flying robot has been developed at DLR. Figure 1.2 shows the two rovers, LRU1 and LRU2 (LRU: *Lightweight Rover Unit*), as well as the micro aerial vehicle (MAV) ARDEA (*Autonomous Robot Design for Extraterrestrial Applications*).
ARDEA is characterized by its high maneuverability which enables it to act as a fast scout easily reaching areas that are otherwise difficult to access such as craters or caves. It is equipped with two pairs of ultra-wide angle stereo camera. Each of these cameras provide inputs to an independent visual odometry [6].
LRU1 and LRU2 both have individually controlled and powered wheels on actuated boogies empowering them to autonomously navigate and explore in rough terrain. They are both equipped with a set of on-board sensors such as different navigation or scientific cameras.
This thesis will cover a specific use case of collaboration between ARDEA and LRU1 who can additionally serve as a carrier for the MAV with a take-off and landing platform. In particular, the following scenario is considered. Both platforms are deployed in a planetary mission and ARDEA is currently on an exploration flight. While being in the field of view of one of LRU1's cameras, it discovers an interesting detail and sends a command to LRU1 demanding for further inspection by the LRU and its instruments. In that case, the orientation and position of ARDEA relative to the rover needs to be estimated as accurately as possible to allow for a safe navigation. The goal of this thesis will be to determine the transformation between the two frames by only relying on on-board sensor information (camera images) of LRU and VO readings of ARDEA.

Figure 1.2.: DLR's mobile robots LRU1 (bottom left), LRU2 (bottom right) and ARDEA (top) during experiments at a Moon-analagoue test site on Mt. Etna, Italy [4].

1.1. Problem Statement

The use case that has just been described can be formulated as a camera pose estimation problem or more precisely, as a modified version of a Perspective-n-Point (PnP) problem. In the classic definition of the PnP problem, the camera pose (orientation and translation) relative to a world reference frame can be determined based on a set of known 3D points and 2D observations. These 3D-2D correspondences are given by a set of known 3D keypoints and the corresponding 2D observations can be detected in a single image frame.

In contrast to the classic definition, we assume the distance between ARDEA and the LRU to be too large to extract a reliable set of multiple keypoints on ARDEA in the image. It is shown in Figure 1.3 that hardly any reliable matches between ORB features [7] can be obtained when ARDEA is relatively small compared to the image dimensions (roughly 50×50 compared to original dimensions of 2452×2056).
In fact, for even larger distances, ARDEA's appearance reduces to only one reliable keypoint, the current position in the frame. Thus, instead of using a set of 3D-2D correspondences in a single image, a sequence of points on a flight trajectory over time can be used to estimate the camera pose. For this modified definition of a temporal PnP problem, 2D observations can be obtained from detecting and tracking ARDEA in a sequence of images. Corresponding 3D points of the trajectory can be inferred from on-board sensor data of ARDEA. Concretely, Visual Odometry (VO) results, consisting of delta orientations and translations together with uncertainty estimates, can yield information about the 3D trajectory.

This leads to a two-part definition of the problem that will be addressed in this thesis.

1. **Tracking of ARDEA**
   To obtain 2D observations of ARDEA's trajectory in the image plane, an approach for detecting and tracking ARDEA throughout a sequence of images needs to be established.

2. **Pose Estimation**
   A modified PnP problem is supposed to be solved based on 3D-2D correspondences describing the flight trajectory of ARDEA in its fixed object frame and in the image plane. Additionally, one goal of this thesis is to incorporate 3D covariance information provided by the VO into pose estimation to improve the results.

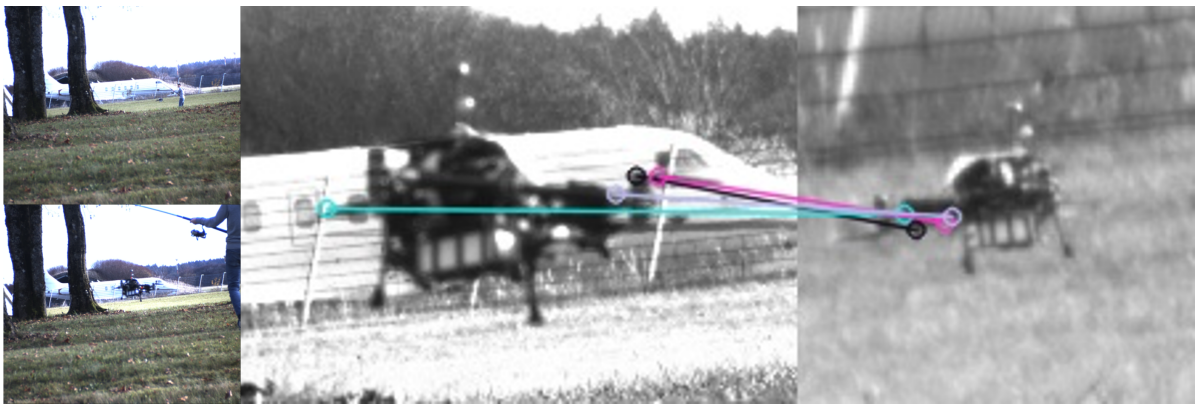


Figure 1.3.: Region of ARDEA is extracted for close range image (bottom left) and far range image (top left). ORB features are computed using the *OpenCV* implementation. Best four matches are highlighted.

1.2. Contributions and Outline

As indicated by the problem definition, the contributions of this thesis will be manifold and will be dealing with both of the sub-problems. The main contributions can be identified as the following components.

- Visual object tracking is one of the key challenges of this work. Two different approaches will be presented to achieve the goal of reliably tracking an object of interest. One of the presented approaches is based on conventional computer vision methods. The other one uses deep-learning based techniques. Both algorithms will be compared in terms of accuracy, reliability and generalization capabilities. Moreover, it will be shown how they can be collaboratively applied to contribute to each other.
- The classic PnP problem will be re-formulated such that the problem is spanned by 3D-2D correspondences over time instead of correspondences in a single image. Furthermore, it will be formulated in terms of VO readings given by delta poses between subsequent frames.
- An existing uncertainty-aware PnP solver (MLPnP [8]) will be extended. The original approach only considers uncertainties of 2D observations. In this thesis, a new approach will be developed based on MLPnP that also takes uncertainties of 3D observations into account.

After this brief introduction, the remainder of this thesis is structured as follows:

Chapter 2 gives a quick overview of noteworthy related work in the two disciplines of visual object detection respectively tracking and pose estimation.
Following that, chapter 3 will deal with the topic of visual object detection and tracking. First, the theoretical background of selected existing tracking or detection methods will be established. Thereafter, the introduced methods will be applied to build a reliable tracker for the concrete use case of this thesis. It will be started with the approach based on conventional methods. After that, a deep-learning based approach will be presented and it will be shown how it can be supported by the previous one.
The theoretical foundations of the pose estimation problem will be covered in chapter 4. In the beginning of the chapter, the problem will be formulated in terms of 2D observations and 3D odometry readings. It will be shown how the problem can be transformed into a reduced observation space as already presented in the MLPnP framework. Furthermore, covariance information will also be propagated into this reduced observation space. Finally, the pose estimation problem is formulated as a minimization problem. Linear as well as non-linear solutions to this optimization problem will be derived incorporating known covariance information.
After the introduction of theoretical concepts in the previous chapters, chapter 5 will evaluate the presented methods in an experimental setup. Experimental data sets that have been recorded for that purpose will be characterized. Both presented tracking approaches that have been introduced are evaluated and compared by qualitative and quantitative means. The proposed novel approach for pose estimation will first be tested in a simulation environment and then applied to real-world data.
Chapter 6 concludes this thesis by summarizing and discussing the key results and will give an outlook on potential improvements and future work.

2
Related Work

As already stated in the introduction, the contributions of this thesis can be split into two sub-problems. One of these problems is tracking an object and its trajectory in a sequence of images which is often referred to as Visual Object Tracking (VOT) in the literature. The other one is pose estimation which is done by solving a modified version of a so-called Perspective-n-Point (PnP) problem. Both of these topics have a long history in research and the most important developments in these areas will be briefly summarized in this chapter.

2.1. Visual Object Tracking

One of the most relevant research topics in the field of computer vision right now is the problem of Visual Object Tracking (VOT) with a large number of new algorithms being proposed every year. The strong interest in VOT can be explained by its large variety of application areas such as human-computer interaction, autonomous driving, robotics or surveillance to name only a few examples.

Despite the great progress that has been achieved in the field of visual tracking within the last years, VOT still remains a very challenging problem facing issues like image noise, complex or fast motions, occlusions, illumination changes or real-time requirements. Although VOT is a very recent topic, it has a long history reaching back several decades. In 2006, an overview of early tracking algorithms was provided in [9]. A more current survey [10] gives a good overview of current state-of-the-art tracking approaches focussing especially on correlation filter based tracking (CFT). The use of deep learning for the task of Multiple Object Tracking (MOT) is summarized in [11]. While task definitions of object tracking might slightly differ between these surveys, they all point out two key steps in every tracking approach: object detection and data association. Hereby, data association describes the process of establishing correspondences between detections of the same object in subsequent frames. Due to the importance of object detection, most common trackers follow the tracking-by-detection approach.
Traditional hand-crafted object detectors, given in [9], are either point-based detectors (e.g. *Harris* [12], *SIFT* [13]), segmentation-based (e.g. *Mean*-Shift [14]) or using background modeling techniques. A common example of the last class has been introduced in [15] and is known as *Mixture-of-Gaussians* (MOG). It sets up a Gaussian Mixture Model (GMM) to distinguish between background and foreground pixels.
While all these methods use different object representations (points, contours, pixel-based), most state-of-the-art approaches rely on the representation of an object by its rectangular bounding box.
For instance, one group of algorithms relying on bounding box representations which has been quite popular in research are *Correlation Filter based Trackers* (CFTs). As already indicated by the name, CFTs use correlation filters trained on target image patches for the purpose of object

tracking. The target position in a new frame can then be estimated by determining the position of the maximum filter response. Initially, the requirement to train those filters in advance made these approaches inappropriate for online tracking. That changed with the development of the *Minimum Output of Sum of Squared Error* (MOSSE) filter [16]. Minimizing the sum of squared errors between an desired output and the actual filter response in the Fourier domain allowed for an efficient online training. Based on MOSSE, a large number of state-of-the-art tracking approaches have been proposed lately.
One of those extensions is known as *Kernelized Correlation Filter* (KCF) [17] tracker which successfully integrated the use of kernel functions into filter training. By that, tracking accuracy could be improved. Still operating in the Fourier domain, KCF was able to keep almost the same time-efficiency as MOSSE. Furthermore, KCF integrated *Histogram of Orientated Gradients* (HOG) [18] features to extend CFTs to multiple channels working on feature representations of the objects. Besides HOG features, other feature representations that are widely applied in CFT approaches are for instance *Colorname* (CN) features [19] or most recently also deep features. The performance of these basic examples of CFTs is limited by the assumption of the object maintaining its original size. Furthermore, their performance degrades in the case of irregularly shaped objects which might result in the CFT learning background information.
There have been several developments to overcome these drawbacks. To address the issue of scale changes, *Discriminative Scale Space Tracking* (DSST) [20] has been proposed which learns separate filters for translation and scale estimation. Moreover, various approaches have been introduced that incorporate measures of regularization to down-weight background information or corrupt examples. Examples are the *Spatially Regularized Discriminative Correlation Filter* (SRDCF) [21] tracker or the *Spatial-Temporal Regularized Correlation Filter* (STRCF) [22] tracker. Another remarkable work has been *Channel Spatial Reliability for Discriminative Correlation Filter* (CSRDCF) [23] which integrates channel and spatial reliability measures into the learning stage of the filter. A probabilistic approach is used to obtain a binary spatial reliability map indicating which pixel values should be ignored. Channel reliability is measured by maximal filter responses per feature channel. All CF based tracking approaches share their computational efficiency and real-time capability.

Not only CFTs have been heavily under investigation, but with the rapid development in the field of deep learning, also great progress has been achieved for object detection. Learning-based detectors are usually categorized as *two-stage* or *one-stage detectors*. The first stage of a two-stage detector generates a set of candidate proposals which are then classified as foreground classes or background in the second stage. In contrast, one-stage detectors formulate the problem as a single regression problem which can be trained end-to-end from raw images to class probabilities and bounding boxes. In general, two-stage detectors usually achieve higher localization and object recognition accuracy whereas one-stage detectors outperform two-stage detectors in terms of inference speed [24].
Algorithms of the so-called R-CNN family can be mentioned here as commonly used two-stage detectors. Original *R-CNN* [25] has been improved over the years resulting in *Fast R-CNN* [26] and *Faster R-CNN* [27] and a large number of further extensions. Despite the great accuracy of R-CNN type detectors, the use case of this thesis demands for real-time algorithms and thus, one-stage detectors are more favorable in this case. The first noteworthy one-stage detectors are probably *SSD* [28] and *YOLO* [29] which has a large number of extensions [30],[31],[32]. However, their accuracy still trailed that of two-stage detectors. The authors of [33] identified class imbalance during training of the network as the main reason limiting the performance of one-stage detectors. This class imbalance is caused by the large number of candidate locations that are evaluated per image. The majority of them are easy negatives, which means they

are not containing any object. Hence, they do not contribute to useful learning signals and can overwhelm training. To address the issue of class imbalance, a novel loss formulation is introduced in [33] focussing on training hard negative samples. This loss is called *Focal Loss* and the network architecture used for training is known as *RetinaNet*.
In the context of this thesis, several of the previously introduced tracking or detection approaches are combined to build a reliable tracker for ARDEA. Conventional methods are applied in simple use cases to generate labeled training data. Using this labeled data, a RetinaNet model will be trained on detection of ARDEA.

2.2. Perspective-n-Point Problem

The goal of the PnP problem is to estimate the pose of a camera (relative orientation and translation) in a world reference frame given a set of n corresponding 3D points in the reference frame and its 2D projections in the image. This task is important in a large number of applications, such as robot localization and object manipulation, photogrammetry, augmented reality or surgical navigation. Driven by this wide range of potential applications, which often require robust, efficient and highly accurate solutions, a great variety of algorithms has been developed addressing the PnP problem.
Research on this topic has a long tradition, starting with the first algorithms beeing introduced in the 1980's. These early developments focused on solving the PnP problem with the minimum number of three given correspondences which yields up to four solutions. Some well-known solutions for the P3P problem are given in [34] or [35]. Additional points can be used for disambiguation as shown for example in [36] where a solution to the P4P respectively P5P problem is presented. Unfortunately, these algorithms are prone to outliers or noisy measurements and thus, often have to be applied in RANSAC [34] based outlier rejection schemes.
Whereas the previously introduced approaches are dependent on a fixed number of points, most solvers can handle an arbitrary number of given correspondences. Following [8], these can be divided into iterative, non-iterative or polynomial, non-polynomial solvers. Iterative solvers aim to minimize different objective functions based on geometric or algebraic errors. Examples of iterative methods are LHM [37] or *Procrustes PnP* (PPnP) [38]. The latter one minimizes the error between the object coordinates and the reprojected image points in the camera frame. High computational costs as well as their sensitivity to local minima are major drawbacks of iterative methods.
Similarly, early non-iterative solvers suffered from high computational complexity. A breakthrough in the development of non-iterative PnP solvers was the introduction of EPnP [39], the first efficient non-iterative solution. It was later improved by subsequent Gauss-Newton minimization [40]. The speed-up of EPnP is achieved by reducing the problem from an arbitrary number of correspondences to finding the position of a fixed number of four control points which are a weighted sum of all 3D points. Most recent, non-iterative state-of-the-art solutions are polynomial solvers replacing linearizations of the EPnP approach with polynomial solvers. Among these are the *Direct Least-Squares* (DLS) [41], the *Accurate and Scalable PnP* (ASPnP) [42], the *Optimal PnP* (OPnP) [43] and the *Unified PnP* (UPnP) [44] algorithm.
Thus far, all algorithms share the assumption of equally accurate observations and the lack of erroneous correspondences which yields geometrical optimality in most cases but not statistical optimality. The first approach to integrate an algebraic outlier rejection criterion into the pose estimation has been introduced as *Robust Efficient Procrustes PnP* (REPPnP) [45]. After eliminating correspondences with an algebraic error exceeding a certain threshold, the final solution is obtained by iteratively solving the Orthogonal Procrustes problem. Using the representation of all points as four control points adapted from the EPnP algorithm, the approach remains

very efficient and achieves speed gains by a factor of 100 compared to RANSAC based outlier rejection.
The first algorithm that explicitly integrates observation uncertainties into the framework is known as *Covariant Efficient Procrustes PnP* (CEPPnP) [46]. Alike REPPnP, it still uses the control point formulation of EPnP. 2D feature uncertainties are assumed to be known and expressed in terms of covariance matrices. The uncertainties are then propagated into the subspace spanned by the control points using the Jacobian. Finally the solution to the PnP problem is obtained from a Maximum Likelihood (ML) minimization approximated by an unconstrained Sampson error function which naturally penalizes noisy correspondences. Furthermore, CEPPnP is able to maintain real-time capability.
Another approach towards solving the PnP problem including 2D feature uncertainties is presented as *Maximum Likelihood PnP* (MLPnP) [8]. In this case, all points and uncertainties are propagated into a reduced observation space which is given by the tangent vector space on the unit sphere in the camera frame and has been derived in [47]. Instead of performing ML minimization to estimate the control points as in CEPPnP, MLPnP directly minimizes residuals in the tangent planes over the unknown quantities, the rotational and translational parameters, without losing real-time capability. This enables MLPnP to provide pose uncertainties which also represent a measure of reliability.
Later, *Image Uncertainty-Based Absolute Camera Pose Estimation with Fibonacci Outlier Elimination* (IUPnP) [48] has been published based on MLPnP. It combines the approach of minimizing residuals in the reduced observation space with an algebraic outlier rejection scheme. Correspondences with tangent space residuals exceeding a threshold are iteratively rejected while also iteratively updating the threshold value using the Fibonacci technique.
However, there are hardly any known PnP solvers that also take observation uncertainties in the 3D reference points into account. To the best of my knowledge, the only work that also considers 3D uncertainties is called EKFPnP (*Extended Kalman Filter for Camera Pose Estimation in a Sequence of Images*) and has recently been published [49]. Uncertainties are incorporated from 2D feature uncertainties and implicitly from the camera motion history. An extended Kalman Filter (EKF) is applied as probabilistic model for camera pose estimation. The pose estimation is done in two steps. First, during prediction, the camera motion model is applied to obtain an a priori pose estimate of the camera pose which is refined in the correction step by minimizing the reprojection error. One advantage of this approach is that it also yields covariance estimates of the resulting pose which can again be taken as a measure of reliability.
This thesis will present a novel approach towards solving the PnP problem with consideration of observation uncertainties in 2D and 3D. The theoretical framework of MLPnP is modified to operate on odometry readings as input. Furthermore, it will be shown how covariances from odometry readings can be propagated into the same reduced observation space. Finally, the pose can be obtained from a generalized least-squares minimization of weighted residuals in the tangent space.

3 Tracking Approach

This work deals with two separate sub-problems. The first part consists of detecting and tracking a certain object (ARDEA) in a sequence of images and determining its trajectory. For this specific task, two different approaches have been implemented which will be presented. One of these approaches is based on conventional methods. The other one is deep learning based. In the beginning of this chapter, the theoretical background of the applied methods is briefly explained. After that, it will be described how these methods are combined to successfully track ARDEA in any video sequence and how to obtain the corresponding 2D trajectory.

3.1. Theoretical Background

Detection and tracking of ARDEA is achieved by relying on maninly three methods. A background subtraction approach called *Mixture of Gaussians* (MOG) [15],[50] is combined with *CSRDCF* [23], a correlation filter based tracker (CFT), to build a conventional tracker. Additionally a *RetinaNet* [33] detector is trained on detection of ARDEA. For all of the three methods the fundamental concepts will be briefly introduced.

3.1.1. Background Subtraction: Mixture of Gaussians (MOG)

Background subtraction methods aim to detect intruding objects in a scene. In many cases a basic assumption is that the scene without the intruding object can be well described by a statistical model. This model is comprised of probability density functions for each pixel $\boldsymbol{x}$ (value in a colorspace, e.g. RGB) separately. The density functions can be learned from a batch of training samples (image frames) $\chi_T = \{\boldsymbol{x}_t, \ldots, \boldsymbol{x}_{t-T}\}$ over a time period T with $\boldsymbol{x}_t$ being the pixel values at a time t. As there can be either samples belonging to foreground (FG) or background (BG) objects in the training batch, a *Gaussian Mixture Model* (GMM) with M components to describe a probability density function for every pixel can be defined as:

$$p\left(\boldsymbol{x}|\chi_T, BG + FG\right) = \sum_{m=1}^{M} \pi_m \mathcal{N}\left(\boldsymbol{x}|\boldsymbol{\mu}_m, \sigma_m^2 \mathbf{I}\right) \tag{3.1}$$

with π_m denoting cluster weights and the cluster means $\boldsymbol{\mu}_m$ respectively variances σ_m^2. The number of componentes M has to be specified in advance in the original approach [15]. Recursive update equations for the parameters of the m'-th component of the GMM for a new sample $\boldsymbol{x}_t$ are given by:

$$\begin{aligned} \pi_m &= \pi_m + \alpha\left(o_m^{(t)} - \pi_m\right) \\ \boldsymbol{\mu}_m &= \boldsymbol{\mu}_m + o_m^{(t)} \frac{\alpha}{\pi_m} \boldsymbol{\delta}_m \\ \sigma_m^2 &= \sigma_m^2 + o_m^{(t)} \frac{\alpha}{\pi_m}\left(\boldsymbol{\delta}_m^T \boldsymbol{\delta}_m - \sigma_m^2\right) \end{aligned} \tag{3.2}$$

where $\boldsymbol{\delta}_m = \boldsymbol{x}_t - \boldsymbol{\mu}_m$, $o_m^{(t)}$ defining the ownership which is 1 for the closest Gaussian and 0 otherwise and a parameter α which sometimes is called a learning rate as it influences the update steps.
Usually, as an intruding object should only be present in a small number of samples due to movement, the foreground objects will be represented by clusters with small weights π_m. Thus, the background model can be approximated by the clusters with the B largest weights:

$$p\left(\boldsymbol{x}|\chi_T, BG\right) = \sum_{m=1}^{B} \pi_m \mathcal{N}\left(\boldsymbol{x}|\boldsymbol{\mu}_m, \sigma_m^2 \mathbf{I}\right) \tag{3.3}$$

Assuming the weights to be in descending order, B can be chosen in the following way:

$$B = \arg\min_b \left(\sum_{m=1}^{b} \pi_m > (1 - c_f) \right) \tag{3.4}$$

where c_f is seen as a measure of the maximum portion of the data belonging to FG objects.
Finally, it can be decided whether a pixel belongs to the background by the following decision rule:

$$p\left(\boldsymbol{x}|\chi_T, BG\right) > c_{thr} \tag{3.5}$$

with a threshold value c_{thr}.

Using this decision rule, one can derive a background model for the input image sequence. Intruding (FG) objects in the current image frame can then be detected by computing and thresholding the absolute difference between the input frame and the background model. As a result, a foreground mask is obtained marking regions of potential objects. The process of object detection using a background subtraction mechanism is illustrated in Figure 3.1.
Zivkovic et al. [50] presented an extended version of the described approach. It is an improvement by automatically selecting the number of required Gaussian components while simultaneously reducing the computation time. A public implementation of the improved approach is available within the *OpenCV* framework.

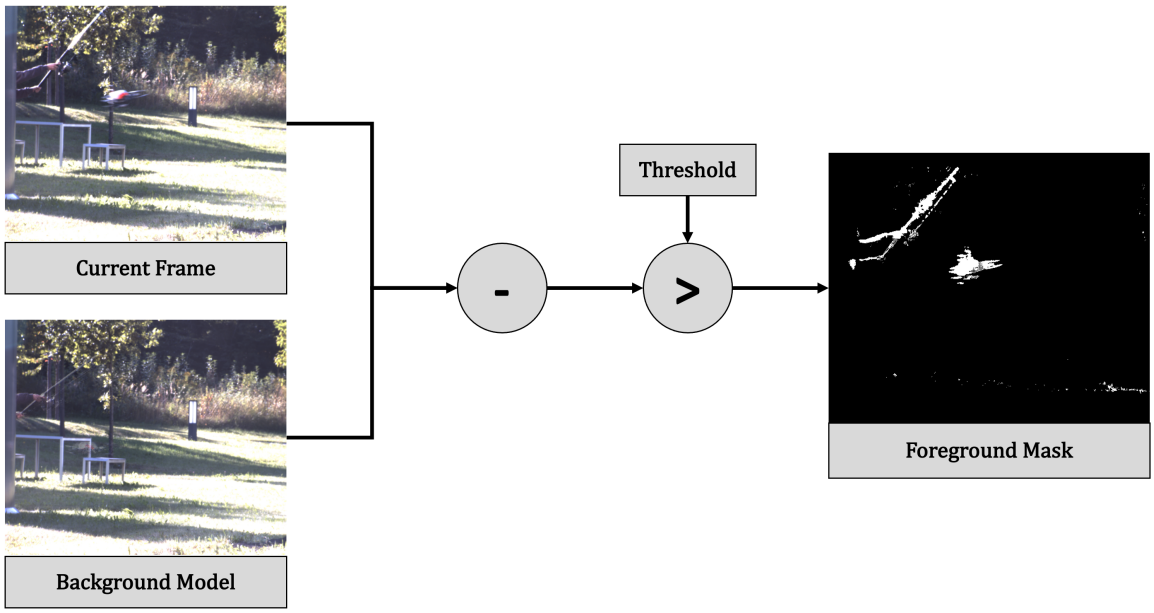


Figure 3.1.: Working Principle of Object Detection using Background Subtraction.

3.1.2. CSRDCF: Discriminative Correlation Filter with Channel and Spatial Reliability

Correlation Filter based tracking methods in general follow the concept of tracking-by-detection. The basic idea is to learn a correlation filter from a training sample which then enables the detection of the same object in a different image. The new position is estimated as the location of the maximum filter response. The general framework is illustrated in Figure 3.2.
For a given image sequence, the filters are initialized from a target patch cropped at the target's location in the initial frame of the sequence. During tracking, the object position in a new frame can be estimated based on the estimated position in the previous frame. Therefore, a feature map in the new frame is extracted at the location of the object in the previous frame. After weighting the feature map with a cosine window to suppress boundary effects, the filter response is calculated in the Fourier domain and transformed back into spatial domain to obtain the response map. The maximum score in this response map then defines the new estimated position of the target. Finally, the estimate of the updated position will be used to update the correlation filters.
Estimating the optimal correlation filters can be seen as the key step of this pipeline. Instead of operating on raw images, usually feature maps are extracted from every input image. The goal can be formulated as estimating the filter $\boldsymbol{h}$ such that it yields a desired output $\boldsymbol{g}$ when applying the filter to the input feature map $\boldsymbol{f}$. This can be stated as a minimization problem:

$$\arg\min_{\boldsymbol{h}} \sum_{d=1}^{N_d} ||\boldsymbol{f}_d * \boldsymbol{h}_d - \boldsymbol{g}||^2 \tag{3.6}$$

where N_d is the number of feature channels and $*$ represents the convolution operation. Most correlation filter based tracking approaches aim to solve that problem in the Fourier Domain instead of solving it in the spatial domain. Applying a Fast Fourier Transform ($\hat{(\cdot)}$ is used to label Fourier domain quantities) drastically reduces the computational cost as the convolution operation becomes element-wise multiplication (denoted by $\odot$).

$$\arg\min_{\boldsymbol{h}} \sum_{d=1}^{N_d} \left|\left|\hat{\boldsymbol{f}}_d \odot \hat{\boldsymbol{h}}_d - \hat{\boldsymbol{g}}\right|\right|^2 \tag{3.7}$$

Many CFT approaches additionally incorporate regularization terms that can reduce the influence of background information. By the use of regularization the tracker can be prevented

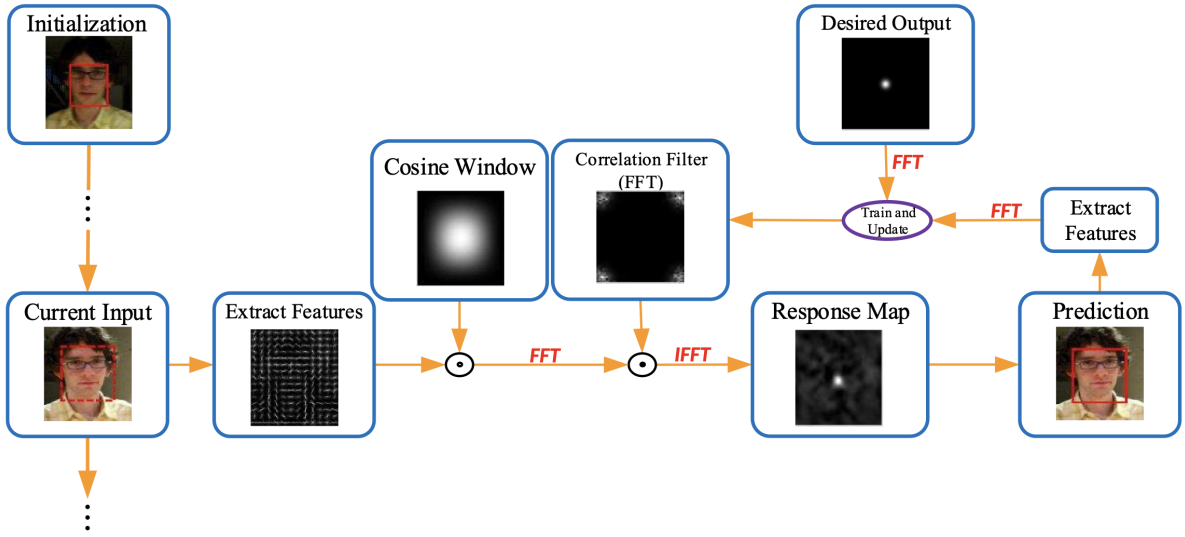


Figure 3.2.: Schematic Overview of a CFT approach [51]

from learning the background of irregularly shaped objects. A spatially regularized version of equation (3.7) with a regularization parameter λ becomes

$$\arg\min_{\boldsymbol{h}} \sum_{d=1}^{N_d} \left\| \hat{\boldsymbol{f}}_d \odot \hat{\boldsymbol{h}}_d - \hat{\boldsymbol{g}} \right\|^2 + \lambda \sum_{d=1}^{N_d} \left\| \hat{\boldsymbol{h}}_d \right\|^2 \tag{3.8}$$

For this minimization problem, there still exists a closed-form solution. However, the solution still suffers from the assumption of all pixels being equally reliable for filter learning. To address that problem, CSRDCF extends common correlation filter based approaches by the use of a binary spatial reliability map $\boldsymbol{m}$ indicating the reliability of each pixel for tracking. This reliability of a pixel $\boldsymbol{x}$ conditioned on its appearance $\boldsymbol{y}$ can be specified as

$$p\left(m = 1 | \boldsymbol{y}, \boldsymbol{x}\right) \propto p\left(\boldsymbol{y} | m = 1, \boldsymbol{x}\right) p\left(\boldsymbol{x} | m = 1\right) p\left(m = 1\right) \tag{3.9}$$

In equation (3.9), the appearance likelihood $p\left(\boldsymbol{y} | m = 1, \boldsymbol{x}\right)$ in this approach is computed from color histograms of the foreground and background. The prior $p\left(m = 1\right)$ is given by the ratio between the foreground and background size and as spatial prior $p\left(\boldsymbol{x} | m = 1\right)$, an Epanechnikov kernel is chosen. Using the Epanechnikov kernel sets larger weights on central pixels which are more likely to contain foreground pixels. A maximum a posteriori solution for the spatial reliability map $\boldsymbol{m}$ is obtained by a Markov random field solver. Finally, $\boldsymbol{m}$ is imposed as additional constraint on a dual problem of (3.8)
Furthermore, CSRDCF not only constructs a spatial reliability map, but it also incorporates channel reliability. Introducing a channel reliability measure is supposed to distinguish filter channels that have high discriminative power from those with low discriminative power. Channel reliability is comprised of a *learning* channel reliability measure on the one hand and a *detection* channel reliability measure on the other hand.
Learning channel reliability is considered during filter learning and expresses how well each channel's response fits the desired response $\boldsymbol{g}$. Thus, the maximum response value $w_d = \zeta \max\left(\boldsymbol{f}_d * \boldsymbol{h}_d\right)$ of every learned filter channel can be used as a reliability measure using a normalization constant ζ such that $\sum_d w_d = 1$.
In contrast to that, detection reliability of a feature channel expresses the reliability of predicting the target's new position as the location of the maximum filter response in the corresponding channel. This reliability can be modeled as the ratio of the second and first major mode in the response map. This can be justified by a simple example. If the two major modes were of similar magnitude, both positions would equally likely be the target's estimated position in the new frame. Hence, the estimated position would be highly uncertain. Otherwise, if the maximum response can be clearly identified, tracking can be achieved with low uncertainties.
Apart from spatial and channel reliability measures, CSRDCF also integrates *Discriminative Scale Space Tracking* (DSST) by *Danelljan et al.* [20] which learns a separate filter for scale estimation.
The implementation of CSRDCF is publicly available within the *OpenCV* framework (where it is called *CSRT*). The original implementation by *Lukezic et al.* uses HOG and CN features and was proven to satisfy real-time requirements. By the time of its publication in 2016 it achieved state-of-the art performance on several benchmarks (e.g. OTB100 or VOT2016).

3.1.3. RetinaNet

RetinaNet is a state-of-the-art one-stage object detector which has been published in 2018. Object detection is defined as a regression problem and the network can be learned end-to-end from input images to output bounding boxes and class probabilities. Its architecture is comprised of a backbone network and two task-specific networks. The overall structure is

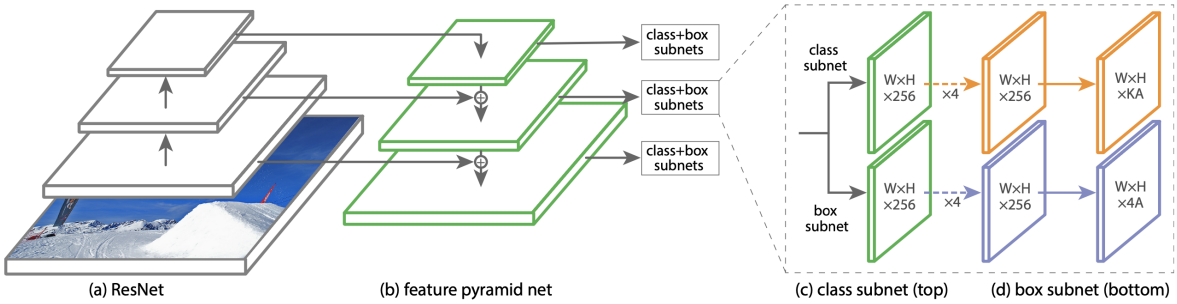


Figure 3.3.: Network architecture of RetinaNet consisting of (b) a Feature Pyramid Network (FPN) on top of a (a) ResNet architecture as backbone with two task-specific subnetworks for (c) classification and (d) bounding box regression [33].

visualized in Figure 3.3.
Combining a ResNet architecture with a Feature Pyramid Network (FPN) [52] in the backbone network enables the network to generate a rich, multi-scale convolutional feature pyramid. To detect objects at different scales, each level of the pyramid can be used. Fixed anchor boxes are defined on each level for every location of the feature map with different sizes and aspect ratios covering the whole image. Each anchor is assigned a one-hot vector of classification targets (length K: number of classes) and a vector of box regression targets (4 elements). The assignemnt is done base on the Intersection over Union (IOU) between anchor boxes and ground-truth object boxes. Each of these pyramidal levels feeds a classification and a bounding box regression feed-forward network.

Box Regression Subnet

The box regression network yields $4A$ (A: number of predefined anchor boxes) linear outputs per spatial location on every level. These outputs predict the relative offset between every anchor and its assigned ground-truth bounding box using the standard box parametrization of R-CNN [25]. The four output parameters can be interpreted as center offsets p_x and p_y respectively width and height offsets p_w and p_h. The localization loss is given by the standard smooth $L1$ loss for predictions $\boldsymbol{p}$ and regression targets $\boldsymbol{t}$ [26]:

$$L_{loc}\left(\boldsymbol{p}, \boldsymbol{t}\right) = \sum_{i \in \{x,y,w,h\}} smooth_{L1}\left(p_i - t_i\right) \tag{3.10}$$

with the regression targets being calculated from anchor box parameters a_i and ground-truth box parameters g_i using

$$\begin{aligned} t_x &= \frac{g_x - a_x}{a_w} \\ t_y &= \frac{g_y - a_y}{a_h} \\ t_w &= \log\left(\frac{g_w}{a_w}\right) \\ t_h &= \log\left(\frac{g_h}{a_h}\right) \end{aligned} \tag{3.11}$$

The smooth $L1$ loss in equation (3.10) is calculated as follows:

$$smooth_{L1}\left(x\right) = \begin{cases} 0.5x^2 & |x| < 1 \\ |x| - 0.5 & |x| \geqslant 1 \end{cases} \tag{3.12}$$

Classification Subnet

In parallel to the box regression subnet, the classification subnet predicts the probability of object presence per spatial location as a one-hot encoded vector. For training the classification net, RetinaNet firstly proposed to use a novel loss formulation, the so-called focal loss. Note that per image, the detector often evaluates several thousands of anchor boxes with only a few of them really containing any objects of interest. Hence, many of them are easy negatives which do not contribute to a useful learning signal. The focal loss is aiming to address this problem of class imbalance by down-weighting easily classified examples and focussing on training on hard examples. Consider a binary classification case with ground-truth classes $y \in \{\pm 1\}$ and $p \in [0, 1]$ being the probability for the class $y = 1$. Using the notation of p_t as

$$p_t = \begin{cases} p & y = 1 \\ 1 - p & \text{otherwise} \end{cases} \tag{3.13}$$

the focal loss can be formulated as follows:

$$FL\left(p_t\right) = -\left(1 - p_t\right)^{\gamma} log\left(p_t\right) \tag{3.14}$$

The formula for the focal loss includes a focusing parameter $\gamma \geqslant 0$ that is used to adjust the rate at which easy examples are down-weighted. In the special case of $\gamma = 0$ the focal reduces to the simple cross entropy loss used in many classification problems.
The total loss function for training is comprised of the sum of focal loss as classification error function and the smooth L1 loss as localisation error.
For prediction on new data, top predictions of all levels are merged and a non-maximum suppression with a threshold of 0.5 on the class probability is applied to yield final outputs.

3.2. Applied tracking algorithms

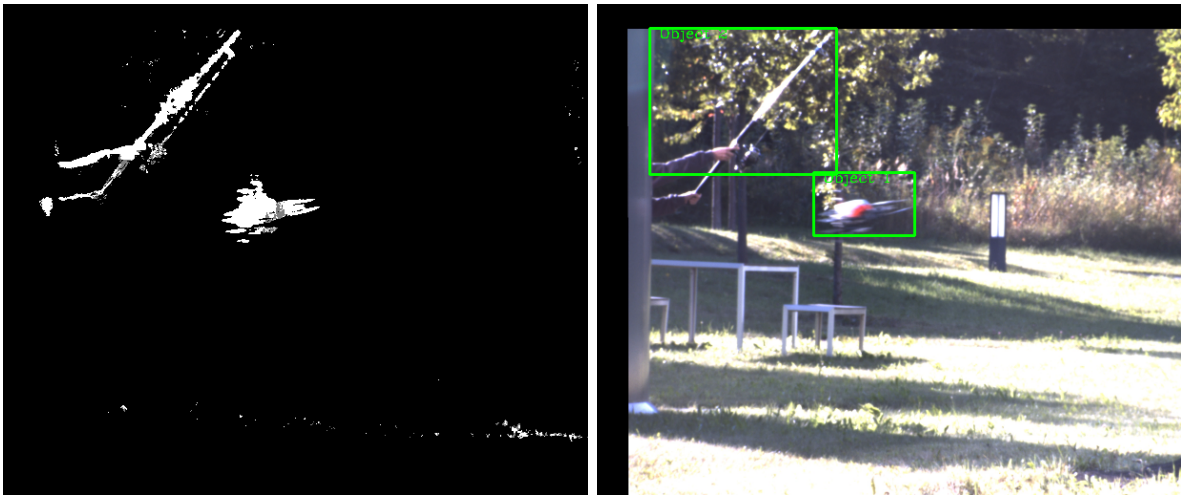
This section will describe how the two conventional methods, MOG and CSRDCF, can be combined to build an object tracker. Furthermore, it will be stated how a RetinaNet object detector can be used in the setting of this thesis to achieve tracking of ARDEA.

3.2.1. Conventional Tracker based on MOG and CFT

The idea of combining a CFT and a background subtraction for the task of object tracking is adapted from [53]. There, a background subtraction method called LOBSTER [54] is used together with a Kernelized Correlation Filter (KCF) tracker. In this thesis, we will build a combined tracker from the previously introduced methods, the Mixture of Gaussians (MOG) and the Discriminative Correlation Filter with Channel and Spatial Reliability (CSRDCF). The background subtraction algorithm mainly is used for object detection and the filter based tracker can support in data association between detection of subsequent frames.
The tracking approach that is implemented can be mainly divided into three stages:

- **Background Learning Stage:**
  Initially, given an image sequence, in the first frame, no information of the background is known. Thus, in the first stage of the tracker, a background of the current scene needs to be learned. Therefore, the first n_{BG} frames, only the MOG model is updated in every step.

- **Object Initialization Stage:**
  After having learnt a model of the background in the first stage, that model can be used to obtain the thresholded foreground mask. Morphological operations, like dilatation, are



(a) Foreground Mask (b) Object Detections

Figure 3.4.: Visualization of object detections for a given foreground mask resulting from background subtraction

applied to this foreground mask to suppress artefacts. Foreground objects are detected by finding contours in the dilated foreground mask. Each object is described by the rectangular bounding box enclosing the contour and the trajectory is described by its center point. The objects which can be found in an exemplary frame from its foreground mask are visualized in Figure 3.4. Note that, here no mechanism is implemented to identify objects. As it can be seen in the example, not only ARDEA is detected, but also the hinge which ARDEA is fixed to. For each of the objects detected that way, a CSRDCF tracker is initialized using the object's bounding box.

- **Object Tracking Stage:**
  With the initialized objects and the corresponding trackers, the goal now is to track these objects in the remaining frames of the video. This is, where the combination of MOG and CSRDCF really comes into play.
  First, in every frame, the background model as well as the tracker are updated. Using the background model, new *Candidate Object Regions* (COR) can be identified applying the same steps as during object detection. Similarly the CFT update yields *Tracker Outputs* (TO) for every tracked object. The overlap of a COR_i and a TO_j can be computed using the *Intersection over Union* (IoU).

$$IoU_{ij} = \frac{COR_i \cap TO_j}{COR_i \cup TO_j} \tag{3.15}$$

  Using the definition of the IoU, every TO_j is assigned its best matching COR_i meaning the candidate region with the highest IoU value. The association is accepted if it exceeds a certain threshold value c_{IoU}, i.e.

$$IoU_{ij} > c_{IoU} \tag{3.16}$$

  . Based on this assignment, several states of the tracking target are defined.

  1. **Tracked:** 1 TO is matched to 1 COR
     This defines the most simple case. Every object can be uniquely assigned to a corresponding candidate region from background subtraction. The COR is used

(a) State: Tracked (b) State: Occlusion

Figure 3.5.: Visualization of the tracking target states. *TO*s are indicated by green bounding boxes, *COR*s by blue bounding boxes. Left: Both *TO*s representing foreground objects can be assigned to distinct *COR*s. Right: Both *TO*s would be assigned to the same *COR* due to occlusion.

to update the track and the tracker is re-initialized. The re-initialization is done to address the issue of long-term drift of the CFT and to prevent the CFT from learning background information. For clarity, this state is shown in Figure 3.5a.

2. **Occlusion:** > 1 *TO*s are matched to 1 *COR*
   This case occurs when two objects are occluding each other. In that case, the background subtraction algorithm will not be able to distinguish the two objects and the *TO* is used to update the track. Figure 3.5b shows an example for the case of occlusion.
3. **Target Lost:** 1 *TO* is not matched to any *COR*
   If the *TO* can not be matched to any *COR*, the track ends and the target is assumed to be lost. Up to this point, the implementation does not enable any re-identification mechanism.

One could extend the approach by the detection of new objects. New objects might be identified from *COR*s that are not overlapping any *TO*. Nevertheless, in the context of this thesis, the presence of the target to be tracked (ARDEA) is assumed from the beginning. The overall described approach for tracking using conventional methods is summarized as Algorithm 1.

3.2.2. Tracking-by-detection using a RetinaNet Detector

The previously introduced tracking approach is able to achieve satisfactory results for very simple scenarios. Prerequisites are a static camera, relatively static backgrounds, a low number of moving objects and relatively constant lighting conditions. If these conditions hold, tuning the hyperparameters of the two conventional tracking approaches can lead to a reliable tracker. Nevertheless, in realistic use cases, these prerequisites cannot be always guaranteed and it is desirable to have a tracking approach that has good generalization capabilities. That means that the tracker is able to perform reliably in arbitrary environments with a minimum of required manual input. That is why in this thesis, it will also be tested whether a learning-based detector can be used for the purpose of tracking. The conventional tracking approach is therefore applied to generate labeled training data (images and corresponding bounding boxes) from simple scenarios. A RetinaNet detector is trained on detection of a single class (ARDEA) using these

labeled images. Additionally, data augmentation has been applied to create additional training data of varying shape and size. So far, only a method for detection of ARDEA is described. However, for the task of tracking one single instance, the problem simplifies drastically as data association can be neglected. It is safe to assume, that the object of interest will only appear once in the scene. That means, if the detection is accurate and fast enough, no tracking would be needed.

Algorithm 1 Conventional Tracker based on MOG and CSRDCF

```
MOG ← Initialize background subtraction model

for Frame f_t in Image Sequence do

    if t < n_BG then                                   ▷ Background Learning Stage
        MOG ← update background model with f_t

    else if t = n_BG then                              ▷ Object Initialization Stage
        MOG ← update background model with f_t
        Objects ← detect objects using BG subtraction
        for every active object in Objects do
            object.Tracker ← initialize CSRDCF tracker for every object
            object.Track ← [ object.center ]
        end for

    else if t > n_BG then                              ▷ Object Tracking Stage
        MOG ← update background model with f_t
        COR ← detect candidate regions using BG subtraction

        for every active object in Objects do
            TO_j ← update object.Tracker
            COR_i ← determine candidate region with maximum IoU
            State ← determine target state based on IoU assignment

            if State = "Tracked" then
                object.Track ← append center of COR_i
                object.Tracker ← re-initialize tracker
            else if State = "Occlusion" then
                object.Track ← append center of TO_j
            else if State = "Lost" then
                object ← set inactive
            end if

        end for

    end if

end for
```

4 Pose Estimation

In this chapter, a novel approach for solving a modified PnP Problem is introduced that explicitly incorporates 2D image uncertainties as well as 3D model uncertainties. Instead of correspondences within one frame, 3D and 2D observations are obtained as trajectories over several frames. Assuming visual odometry readings and image observations as inputs, a problem formulation will be derived and it is shown how corresponding uncertainties can be propagated. Based on [8], the problem will then be transformed into a reduced observation space which is given by the vector tangent space on the unit sphere in the camera frame. Finally, the pose estimation is formulated as a minimization problem and different solution techniques are presented. The overall algorithm is summarized in Algorithm 3.

4.1. Problem Formulation

The main goal is to estimate the current position and orientation of ARDEA relative to the LRU where at a time i the current position is given by the translation vector ${}^{c}\boldsymbol{t}_i \in \mathbb{R}^3$ and the orientation can be represented by a rotation matrix ${}^{c}\mathbf{R}_i \in \mathbb{R}^{3\times3}$. As an input the following quantities are given:

- **Delta Poses from Visual Odometry**
  - Sequence of n odometry readings $\left({}^{i-j-1}\mathbf{R}_{i-j}, {}^{i-j-1}\boldsymbol{t}_{i-j}\right)$ for $j = 0, \ldots, n-1$
    - Relative translation ${}^{i-j-1}\boldsymbol{t}_{i-j} \in \mathbb{R}^3$: position of ARDEA at time $i-j$ with respect to previous frame $i-j-1$
    - Relative orientation ${}^{i-j-1}\mathbf{R}_{i-j} \in \mathbb{R}^{3\times3}$: orientation of ARDEA at time $i-j$ with respect to previous frame $i-j-1$
  - Corresponding uncertainties as covariance matrices $\boldsymbol{\Sigma}_{\delta\boldsymbol{u}_{i-j}\delta\boldsymbol{u}_{i-j}} \in \mathbb{R}^{6\times6}$ (translation and rotation error per axis)
- **2D Observations from Tracking**
  - $n+1$ corresponding 2D image observations $\boldsymbol{x}'_{i-j} \in \mathbb{R}^2$ for $j = 0, \ldots, n$
  - Corresponding uncertainties as covariance matrices $\boldsymbol{\Sigma}_{\boldsymbol{x}'_{i-j}\boldsymbol{x}'_{i-j}} \in \mathbb{R}^{2\times2}$
- Camera intrinsics matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$

The pose estimation problem and the involved geometric transformations are visualized in Figure 4.1.

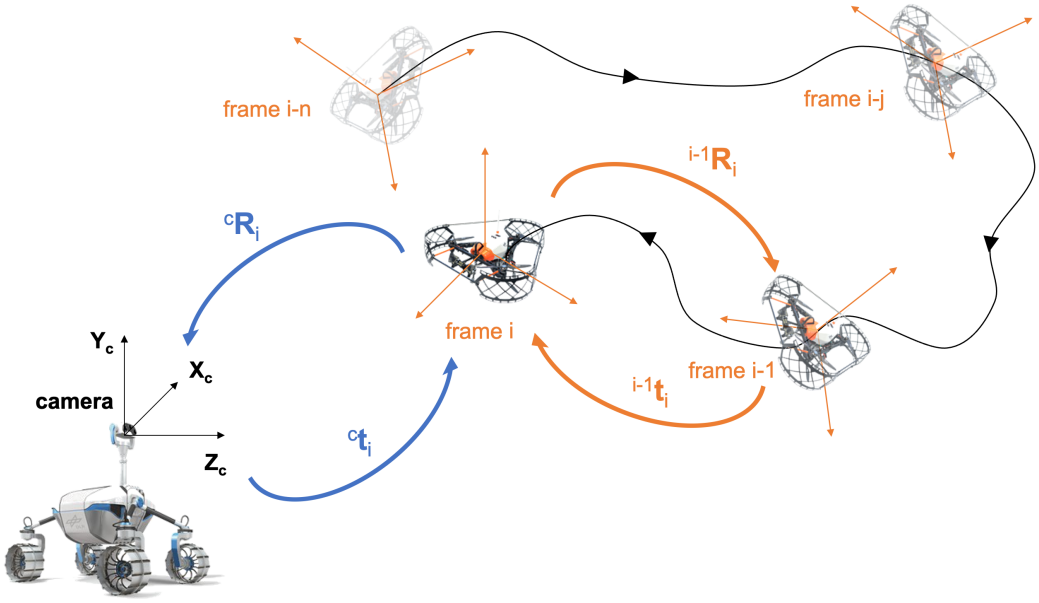


Figure 4.1.: Visualization of the geometric transformations occuring in the pose estimation problem.

4.1.1. Integration of Odometry Readings

At a specific time step $t = i$, the current position of ARDEA in the camera frame ${}^{c}\boldsymbol{p}_i$ is simply given by ${}^{c}\boldsymbol{t}_i$. Based on the known sequence of n odometry readings, the previous positions of ARDEA can be calculated by

$$\begin{aligned}
{}^{c}\boldsymbol{p}_i &= {}^{c}\boldsymbol{t}_i \\
{}^{c}\boldsymbol{p}_{i-1} &= {}^{c}\boldsymbol{t}_i + {}^{c}\mathbf{R}_i{}^{i}\boldsymbol{t}_{i-1} \\
{}^{c}\boldsymbol{p}_{i-2} &= {}^{c}\boldsymbol{t}_i + {}^{c}\mathbf{R}_i{}^{i}\boldsymbol{t}_{i-1} + {}^{c}\mathbf{R}_i{}^{i}\mathbf{R}_{i-1}{}^{i-1}\boldsymbol{t}_{i-2} \\
&\vdots \\
{}^{c}\boldsymbol{p}_{i-n} &= {}^{c}\boldsymbol{t}_i + {}^{c}\mathbf{R}_i{}^{i}\boldsymbol{t}_{i-1} + \ldots + {}^{c}\mathbf{R}_i \prod_{k=1}^{n-1} \left({}^{i-k+1}\mathbf{R}_{i-k}{}^{i-n+1}\boldsymbol{t}_{i-n} \right)
\end{aligned} \tag{4.1}$$

One can observe, that not the odometry readings ${}^{i-j-1}\mathbf{R}_{i-j}$, ${}^{i-j-1}\boldsymbol{t}_{i-j}$ itself occur in these equations but its reversed versions which can be easily converted into one another by

$$\begin{aligned}
{}^{i-j}\mathbf{R}_{i-j-1} &= {}^{i-j-1}\mathbf{R}_{i-j}{}^{T} \\
{}^{i-j}\boldsymbol{t}_{i-j-1} &= -{}^{i-j}\mathbf{R}_{i-j-1}{}^{i-j-1}\boldsymbol{t}_{i-j}
\end{aligned} \tag{4.2}$$

using the fact that the rotation matrix is an orthonormal matrix.
Equation (4.1) can be generalized such that at a time step i, a previous position of ARDEA from j time steps ago can be written as

$${}^{c}\boldsymbol{p}_{i-j} = {}^{c}\mathbf{R}_i{}^{i}\hat{\boldsymbol{p}}_j + {}^{c}\,\boldsymbol{t}_i \tag{4.3}$$

with the newly introduced ${}^{i}\hat{\boldsymbol{p}}_j$ being constructed only from odometry readings as follows:

$${}^{i}\hat{\boldsymbol{p}}_j = \sum_{m=1}^{j} \left(\prod_{n=1}^{m-1} {}^{i-m+n+1}\mathbf{R}_{i-m+n} \right) {}^{i-m+1}\boldsymbol{t}_{i-m} \tag{4.4}$$

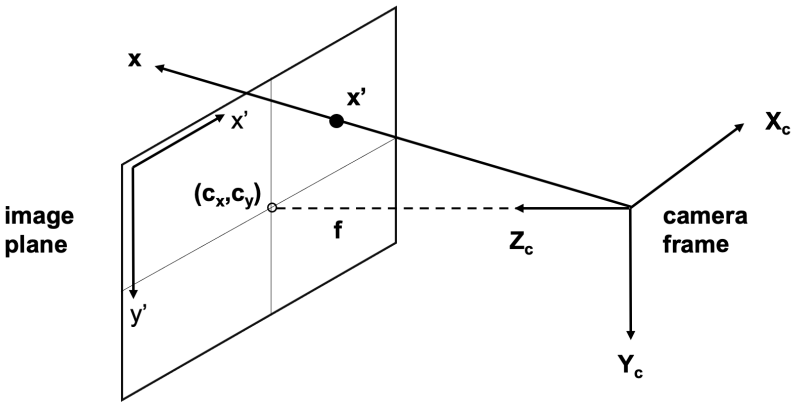


Figure 4.2.: Simplified model of the pinhole projection assuming equivalent focal length in both directions and no skew. A point $\boldsymbol{x}$ in the camera frame is projected onto $\boldsymbol{x}'$ in the image plane.

For efficient computations, this new quantity can also be computed iteratively when sequentially processing odometry readings. This iterative update is given by

$$\begin{aligned} {}^{i}\hat{\boldsymbol{p}}_j &= {}^{i}\hat{\boldsymbol{p}}_{j-1} + \left(\prod_{n=1}^{j-1} {}^{i-j+n+1}\mathbf{R}_{i-j+n}\right) {}^{i-j+1}\boldsymbol{t}_{i-j} \\ &= {}^{i}\hat{\boldsymbol{p}}_{j-1} + {}^{i}\mathbf{R}_{i-j+1}\,{}^{i-j+1}\boldsymbol{t}_{i-j} \\ &= {}^{i}\hat{\boldsymbol{p}}_{j-1} + \Delta\hat{\boldsymbol{p}}_{j-1,j} \end{aligned} \tag{4.5}$$

4.1.2. Application of the Camera Model

In the context of this thesis, the camera is assumed to be calibrated and thus, camera intrinsic parameters are known from calibration. Using the pinhole camera model, 3D points $\boldsymbol{x}_{i-j} \in \mathbb{R}^3$ in the camera coordinate frame corresponding to 2D image observations $\boldsymbol{x}'_{i-j} \in \mathbb{R}^2$ can be obtained up to an unknown distance Z_{i-j} by the inverse pinhole projection. Therefore, the representation of the camera intrinsics as projection matrix $\mathbf{K}$ is used.

$$Z_{i-j}\boldsymbol{x}_{i-j} = \mathbf{K}^{-1}\begin{bmatrix}\boldsymbol{x}'_{i-j}\\ 1\end{bmatrix} \tag{4.6}$$

The model of the projective transformation used in the pinhole camera model is shown in Figure 4.2 and the corresponding projection matrix can be formed from the camera parameters including the focal lengths f_x and f_y in x- respectively y-direction, the optical center coordinates c_x and c_y and a skew parameter s [55].

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$

A subsequent spherical normalization yields general observations lying on the surface of the unit sphere in the camera frame. The resulting observations are denoted as $\boldsymbol{v}_{i-j} \in \mathbb{R}^3$ with $||\boldsymbol{v}_{i-j}|| = 1$.

$$\boldsymbol{v}_{i-j} = \frac{\boldsymbol{x}_{i-j}}{||\boldsymbol{x}_{i-j}||} \tag{4.8}$$

Plugging the inverse projection (4.6) and spherical normalization (4.8) into equation (4.3) for the calculation of the position of ARDEA at an arbitrary time j, yields a general observation equation with unknown scale factors λ_{i-j}.

$$\lambda_{i-j}\boldsymbol{v}_{i-j} = {}^{c}\mathbf{R}_i\,{}^{i}\hat{\boldsymbol{p}}_j + {}^{c}\,\boldsymbol{t}_i \tag{4.9}$$

4.1.3. Covariance Propagation

This section shows how covariances corresponding to 2D and 3D observations can be propagated during transformation of the problem to the camera frame and the following spherical normalization.

A general formula for linear propagation of uncertainties undergoing arbitrary transformations $\pi : \boldsymbol{x} \rightarrow \boldsymbol{y} = \pi(\boldsymbol{x})$ is using the Jacobian $\mathbf{J}_\pi$ of the transformation and is often referred to as *error propagation formula*. It is given by

$$\boldsymbol{\Sigma}_{\boldsymbol{yy}} = \mathbf{J}_\pi \boldsymbol{\Sigma}_{\boldsymbol{xx}} \mathbf{J}_\pi^T \tag{4.10}$$

2D Covariances

For the inverse projection of 2D observations into the camera frame, the Jacobian of (4.6) for covariance propagation is given by the inverse of the camera matrix. The resulting camera frame covariances are obtained as

$$\boldsymbol{\Sigma}_{\boldsymbol{x}_{i-j}\boldsymbol{x}_{i-j}} = \mathbf{K}^{-1} \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{x}'_{i-j}\boldsymbol{x}'_{i-j}} & 0 \\ \mathbf{0}^T & 0 \end{bmatrix} \mathbf{K}^{-T} \tag{4.11}$$

For the following spherical normalization, covariances can be propagated using [47]:

$$\boldsymbol{\Sigma}_{\boldsymbol{v}_{i-j}\boldsymbol{v}_{i-j}} = \mathbf{J}_{\boldsymbol{v}_{i-j}} \boldsymbol{\Sigma}_{\boldsymbol{x}_{i-j}\boldsymbol{x}_{i-j}} \mathbf{J}_{\boldsymbol{v}_{i-j}}^T \quad \mathbf{J}_{\boldsymbol{v}_{i-j}} = \frac{1}{||\boldsymbol{x}_{i-j}||} \left(\mathbf{I} - \boldsymbol{v}_{i-j}\boldsymbol{v}_{i-j}^T\right) \tag{4.12}$$

An overall covariance matrix ${}^{(2D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}}$ can be formed as a block diagonal matrix with elements $\boldsymbol{\Sigma}_{\boldsymbol{v}_{i-j}\boldsymbol{v}_{i-j}}$ on the main diagonal.

$${}^{(2D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}} = \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{v}_i\boldsymbol{v}_i} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \boldsymbol{\Sigma}_{\boldsymbol{v}_{i-n}\boldsymbol{v}_{i-n}} \end{bmatrix} \tag{4.13}$$

The covariance matrices $\boldsymbol{\Sigma}_{\boldsymbol{v}_i\boldsymbol{v}_i}$ propagated in the described way will be singular and thus, also ${}^{(2D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}}$ will be singular. That is the reason why in the original work [8], the reduced observation space is used which will be derived later.

3D Covariances

In the case of 3D pose uncertainties, covariance propagation becomes more complex. This is due to the fact that subsequent measurement errors are not any longer independent. In fact, the errors are summing up over time. Thus, we cannot construct separate covariances matrices $\boldsymbol{\Sigma}_{\boldsymbol{v}_i\boldsymbol{v}_i} \in \mathbb{R}^{3\times3}$ per observation. Instead, for $n+1$ observations, an overall covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{vv}} \in \mathbb{R}^{3(n+1)\times3(n+1)}$ needs to be constructed to appropriately model the dependency between measurements. Therefore, the following error model for translational and rotational errors is introduced

$$\begin{aligned} {}^c\mathbf{R}_i &= {}^c\hat{\mathbf{R}}_i \, {}^c\delta\mathbf{R}_i \\ {}^c\boldsymbol{t}_i &= {}^c\hat{\boldsymbol{t}}_i - {}^c\delta\boldsymbol{t}_i \end{aligned} \tag{4.14}$$

where ${}^c\mathbf{R}_i, {}^c\boldsymbol{t}_i$ are true, but unknown quantities, ${}^c\hat{\mathbf{R}}_i, {}^c\hat{\boldsymbol{t}}_i$ are estimated and ${}^c\delta\mathbf{R}_i, {}^c\delta\boldsymbol{t}_i$ are corresponding error quantities. The rotational errors are locally defined and represented in the matrix ${}^c\delta\mathbf{R}_i$.

$${}^c\delta\mathbf{R}_i = \mathrm{e}^{\lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times} = I + \lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times + \frac{1}{2}\lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times^2 + \frac{1}{3!}\lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times^3 + \dots \tag{4.15}$$

The rotation vector ${}^c\delta\boldsymbol{\phi}_i \in \mathbb{R}^3$ encodes rotation errors. A linear approximation of the matrix ${}^c\delta\mathbf{R}_i$ is used for covariance propagation.

$$ {}^c\delta\mathbf{R}_i \approx \mathbf{I} + \lfloor\, {}^c\delta\boldsymbol{\phi}_i \,\rfloor_\times \tag{4.16} $$

After substituting the error model into the equation for ARDEA's position over time (4.1), one obtains the following linear error propagation:

$$ \begin{aligned} {}^c\delta\boldsymbol{p}_i &= {}^c\delta\boldsymbol{t}_i \\ {}^c\delta\boldsymbol{p}_{i-1} &= {}^c\delta\boldsymbol{t}_i + {}^c\hat{\mathbf{R}}_i \lfloor\, {}^i\hat{\boldsymbol{t}}_{i-1} \,\rfloor_\times^c \delta\boldsymbol{\phi}_i + {}^c\hat{\mathbf{R}}_i {}^i\delta\boldsymbol{t}_{i-1} \\ {}^c\delta\boldsymbol{p}_{i-2} &= {}^c\delta\boldsymbol{t}_i + {}^c\hat{\mathbf{R}}_i \lfloor\, {}^i\hat{\boldsymbol{t}}_{i-1} + {}^i\hat{\mathbf{R}}_{i-1} {}^{i-1}\hat{\boldsymbol{t}}_{i-2} \,\rfloor_\times {}^c\delta\boldsymbol{\phi}_i + {}^c\hat{\mathbf{R}}_i {}^i\delta\boldsymbol{t}_{i-1} \\ &\quad + {}^c\hat{\mathbf{R}}_i {}^i\hat{\mathbf{R}}_{i-1} \lfloor\, {}^{i-1}\hat{\boldsymbol{t}}_{i-2} \,\rfloor_\times^i \delta\boldsymbol{\phi}_{i-1} + {}^c\hat{\mathbf{R}}_i {}^i\hat{\mathbf{R}}_{i-1} {}^{i-1}\delta\boldsymbol{t}_{i-2} \\ &\vdots \\ {}^c\delta\boldsymbol{p}_{i-n} &= \dots \end{aligned} \tag{4.17} $$

A detailed derivation of the error propagation can be found in Appendix A.1.
This can be rewritten in Matrix-Vector form as:

$$ \begin{bmatrix} {}^c\delta\boldsymbol{p}_i \\ {}^c\delta\boldsymbol{p}_{i-1} \\ {}^c\delta\boldsymbol{p}_{i-2} \\ \vdots \\ {}^c\delta\boldsymbol{p}_{i-n} \end{bmatrix} = \mathbf{C} \begin{bmatrix} {}^c\delta\boldsymbol{t}_i \\ {}^c\delta\boldsymbol{\phi}_i \end{bmatrix} + \mathbf{B} \begin{bmatrix} {}^i\delta\boldsymbol{t}_{i-1} \\ {}^i\delta\boldsymbol{\phi}_{i-1} \\ {}^{i-1}\delta\boldsymbol{t}_{i-2} \\ {}^{i-1}\delta\boldsymbol{\phi}_{i-2} \\ \vdots \\ {}^{i-n+1}\delta\boldsymbol{t}_{i-n} \\ {}^{i-n+1}\delta\boldsymbol{\phi}_{i-n} \end{bmatrix} $$

$$ \delta\boldsymbol{p} = \mathbf{C}\delta\boldsymbol{x} + \mathbf{B}\delta\boldsymbol{u} \tag{4.18} $$

with $\mathbf{C} \in \mathbb{R}^{3(n+1)\times 6}$ and $\mathbf{B} \in \mathbb{R}^{3(n+1)\times 6n}$. The construction of the matrices $\mathbf{C}$ and $\mathbf{B}$ are shown in Appendix A.2.
Using this result, the covariance matrices can be propagated according to the following equation:

$$ \boldsymbol{\Sigma}_{\delta\boldsymbol{p}\delta\boldsymbol{p}} = \mathbf{C}\boldsymbol{\Sigma}_{\delta\boldsymbol{x}\delta\boldsymbol{x}}\mathbf{C}^T + \mathbf{B}\boldsymbol{\Sigma}_{\delta\boldsymbol{u}\delta\boldsymbol{u}}\mathbf{B}^T \tag{4.19} $$

In a filtering approach, equation (4.19) can be used directly. For the pose estimation, assuming only delta pose uncertainties given, it reduces to

$$ \boldsymbol{\Sigma}_{\delta\boldsymbol{p}\delta\boldsymbol{p}} = \mathbf{B}\boldsymbol{\Sigma}_{\delta\boldsymbol{u}\delta\boldsymbol{u}}\mathbf{B}^T \tag{4.20} $$

where the overall pose covariance matrix $\boldsymbol{\Sigma}_{\delta\boldsymbol{u}\delta\boldsymbol{u}} \in \mathbb{R}^{6n\times 6n}$ is a block-diagonal matrix containing single pose covariance matrices $\boldsymbol{\Sigma}_{\delta\boldsymbol{u}_i\delta\boldsymbol{u}_i} \in \mathbb{R}^{6\times 6}$ on the main diagonal. Due to the dependency of errors on the whole sequence of measurements, the resulting covariance matrix $\boldsymbol{\Sigma}_{\delta\boldsymbol{p}\delta\boldsymbol{p}} \in \mathbb{R}^{6n\times 6n}$ will be fully occupied. Similarly to the 2D case, a subsequent normalization has to be applied. Therefore, it helps to investigate the iterative structure of the position equations (4.3) - (4.5). From that, it follows that any observation $\boldsymbol{v}_{i-j}$ always only depends on more recent observations, not on older ones. Thus, constructing the Jacobian for the speherical normalization will result in a lower triangular matrix of the following form:

$$ \mathbf{J}_{\boldsymbol{v}} = \begin{bmatrix} \mathbf{J}_{\boldsymbol{v}_i} & & & & \mathbf{0} \\ \mathbf{J}_{\boldsymbol{v}_{i-1}} & \mathbf{J}_{\boldsymbol{v}_{i-1}} & & & \\ \mathbf{J}_{\boldsymbol{v}_{i-2}} & \mathbf{J}_{\boldsymbol{v}_{i-2}} & \mathbf{J}_{\boldsymbol{v}_{i-2}} & & \\ \vdots & \vdots & \vdots & \ddots & \\ \mathbf{J}_{\boldsymbol{v}_{i-n}} & \dots & \dots & \dots & \mathbf{J}_{\boldsymbol{v}_{i-n}} \end{bmatrix} \tag{4.21} $$

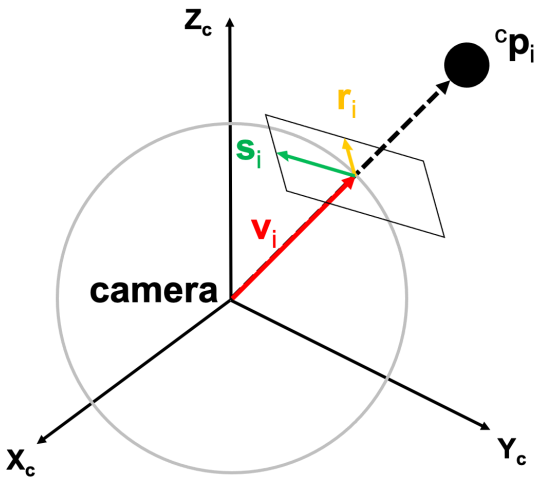


Figure 4.3.: Illustration of the reduced observation space. A point in the camera frame ${}^{c}\boldsymbol{p}_i$ is projected onto the unit sphere. The tangential plane corresponding to projected observation $\boldsymbol{v}_i$ is spanned by its nullspace vectors $\boldsymbol{r}_i$ and $\boldsymbol{s}_i$.

using the Jacobians for single observations from (4.12). A detailed derivation of this result can be found in Appendix A.3.
The resulting overall covariance matrix for 3D observations after normalization is obtained as

$$ {}^{(3D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}} = \mathbf{J}_{\boldsymbol{v}} \boldsymbol{\Sigma}_{\delta \boldsymbol{p} \delta \boldsymbol{p}} \mathbf{J}_{\boldsymbol{v}}^{T} \tag{4.22} $$

Combined Covariance Matrix

Now, after all given uncertainties have been propagated to the unit sphere in the camera frame, we can add the resulting covariance matrices ${}^{(3D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}}$, derived from ARDEA's delta pose inputs, and ${}^{(2D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}}$, derived from image observations, to construct a combined, overall covariance matrix for all observations $\boldsymbol{\Sigma}_{\boldsymbol{vv}}$.

$$ \boldsymbol{\Sigma}_{\boldsymbol{vv}} = {}^{(3D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}} + {}^{(2D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}} \tag{4.23} $$

4.1.4. Reduced Observation Space

Finally, following [8] and [47], the problem is now transformed to a two-dimensional reduced observation space, which is given by tangent planes on the unit sphere. Figure 4.3 illustrates this reduced observation space. For every single observation $\boldsymbol{v}_{i-j}$ for $j = 0, \ldots, n$, the corresponding subspace is spanned by its null space vectors $\boldsymbol{r}_{i-j}$ and $\boldsymbol{s}_{i-j}$.

$$ \mathbf{J}_{\boldsymbol{v}_{r,i-j}} = null\left(\boldsymbol{v}_{i-j}^{T}\right) = \begin{bmatrix} \boldsymbol{r}_{i-j} & \boldsymbol{s}_{i-j} \end{bmatrix} \tag{4.24} $$

The function $null(\cdot)$ computes the null space of a vector by calculating its Singular Value Decomposition (SVD) and taking those right singular vectors corresponding to the two zero singular values. $\mathbf{J}_{\boldsymbol{v}_{r,i-j}}$ can be used to project every observation to its reduced observation space where its reduced observation $\boldsymbol{v}_{r,i-j}$ should be in the origin of the tangent plane. This can be formalized as

$$ \boldsymbol{v}_{r,i-j} = \begin{bmatrix} dr_{i-j} \\ ds_{i-j} \end{bmatrix} = \mathbf{J}_{\boldsymbol{v}_{r,i-j}}^{T} \boldsymbol{v}_{i-j} = \mathbf{0} \tag{4.25} $$

with residual components dr_{i-j} and ds_{i-j}. Applying this projection to our pose estimation problem (4.9) results in

$$ \begin{bmatrix} dr_{i-j} \\ ds_{i-j} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}_{i-j}^{T} \\ \boldsymbol{s}_{i-j}^{T} \end{bmatrix} \lambda_{i-j}^{-1} \left({}^{c}\mathbf{R}_{i} \, {}^{i}\hat{\boldsymbol{p}}_{j} + {}^{c}\boldsymbol{t}_{i} \right) \tag{4.26} $$

with depth values $\lambda_{i-j} \neq 0$. In (4.26), it is assumed that the projection of ${}^{i}\hat{\boldsymbol{p}}_j$ into the reduced tangent space results in the same reduced coordinates as the projection of the 2D observation $\boldsymbol{v}_{i-j}$ and thus, the residual equation should hold.
$\mathbf{J}^T_{\boldsymbol{v}_{r,i-j}}$ is also the Jacobian of the nullspace projection and can be used for covariance propagation. But, as individual observations cannot be considered independent, covariance propagation needs to take into account all observations in all reduced spaces. Therefore the residuals are stacked and the propagation can be obtained as

$$\boldsymbol{\Sigma}_{\boldsymbol{v}_r\boldsymbol{v}_r} = \mathbf{J}^T_{\boldsymbol{v}_r}\boldsymbol{\Sigma}_{\boldsymbol{v}\boldsymbol{v}}\mathbf{J}_{\boldsymbol{v}_r} \tag{4.27}$$

where the overall nullspace Jacobian $\mathbf{J}_{\boldsymbol{v}_r}$ is given by the following block matrix:

$$\mathbf{J}_{\boldsymbol{v}_r} = \begin{bmatrix} \mathbf{J}_{\boldsymbol{v}_{r,i}} & & & \mathbf{0} \\ & \mathbf{J}_{\boldsymbol{v}_{r,i-1}} & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{J}_{\boldsymbol{v}_{r,i-n}} \end{bmatrix} \tag{4.28}$$

$\mathbf{J}_{\boldsymbol{v}_r}$ will be of size $\mathbb{R}^{3(n+1)\times 2(n+1)}$ with blocks being formed by single observation nullspace projections $\mathbf{J}_{\boldsymbol{v}_{r,i-j}} \in \mathbb{R}^{3\times 2}$.

4.2. Pose Estimation as Minimization Problem

Using the representation of all quantities in the previously introduced reduced observation space, the pose estimation can be formulated as a minimization problem. This will be shown in the following section and linear solutions as well as non-linear estimates will be derived.

4.2.1. Derivation of Minimization Problem

Expanding and stacking the residuals in equation (4.26) yields two equations per observation:

$$\begin{aligned} 0 = \quad & r_1\left(\hat{r}_{11}p_1 + \hat{r}_{12}p_2 + \hat{r}_{13}p_3 + \hat{t}_x\right) \\ & + r_2\left(\hat{r}_{21}p_1 + \hat{r}_{22}p_2 + \hat{r}_{23}p_3 + \hat{t}_y\right) \\ & + r_3\left(\hat{r}_{31}p_1 + \hat{r}_{32}p_2 + \hat{r}_{33}p_3 + \hat{t}_z\right) \\ 0 = \quad & s_1\left(\hat{r}_{11}p_1 + \hat{r}_{12}p_2 + \hat{r}_{13}p_3 + \hat{t}_x\right) \\ & + s_2\left(\hat{r}_{21}p_1 + \hat{r}_{22}p_2 + \hat{r}_{23}p_3 + \hat{t}_y\right) \\ & + s_3\left(\hat{r}_{31}p_1 + \hat{r}_{32}p_2 + \hat{r}_{33}p_3 + \hat{t}_z\right) \end{aligned} \tag{4.29}$$

in which $r_{1,2,3}$ and $s_{1,2,3}$ are the elements of the nullspace vectors $\boldsymbol{r}_{i-j}$ respectively $\boldsymbol{s}_{i-j}$ and $p_{1,2,3}$ the elements of ${}^{i}\hat{\boldsymbol{p}}_j$. Introducing the parameter vector $\boldsymbol{u} \in \mathbb{R}^{12}$ containing the components of the estimated rotation matrix ${}^{c}\hat{\mathbf{R}}_i$ and estimated translation vector ${}^{c}\hat{\boldsymbol{t}}_i$ and stacking residuals for n observations yields a homogeneous system of linear equations:

$$\begin{aligned} \mathbf{A}\boldsymbol{u} &= \mathbf{0} \\ \boldsymbol{u} &= \left[\hat{r}_{11}, \hat{r}_{12}, \hat{r}_{13}, \hat{r}_{21}, \hat{r}_{22}, \hat{r}_{23}, \hat{r}_{31}, \hat{r}_{31}, \hat{r}_{32}, \hat{r}_{33}, \hat{t}_x, \hat{t}_y, \hat{t}_z\right]^T \end{aligned} \tag{4.30}$$

in which $\mathbf{A} \in \mathbb{R}^{2(n+1)\times 12}$ represents a design matrix constructed from the coefficients in (4.29). As the parameter vector $\boldsymbol{u}$ includes twelve unknowns, a minimum number of $n + 1 = 6$ observations (for n odometry readings) is required to obtain a solution to the problem.
Alternatively, (4.30) can be reformulated as a minimization problem in a least-squares sense, finding $\boldsymbol{u}^*$ such that

$$\boldsymbol{u}^* = \arg\min_{\boldsymbol{u}} ||\mathbf{A}\boldsymbol{u}||_2^2 \quad \text{subject to } ||\boldsymbol{u}||_2 = 1 \tag{4.31}$$

The constraint $||\boldsymbol{u}||_2 = 1$ avoids retrieving the trivial solution $\boldsymbol{u} = \mathbf{0}$.

4.2.2. Linear Least-Squares Solutions

In this section, different ways to obtain a solution to the previously derived minimization problem are provided, either including or neglecting weighting by known observation uncertainties. A general form of the weighted minimization problem can be found as

$$\min_{\boldsymbol{u}} ||\mathbf{A}\boldsymbol{u}||^2_{\mathbf{W}} \tag{4.32}$$

with a weighting matrix $\mathbf{W}$. In the unweighted case ($\mathbf{W} = \mathbf{I}$), this reduces to the original least-squares problem (4.31). In the weighted case, the inverse of the covariance matrix is used as weighting matrix ($\mathbf{W} = \mathbf{\Sigma}^{-1}_{\boldsymbol{v}_r\boldsymbol{v}_r}$). For a better readability, we will use $\mathbf{\Sigma}^{-1} = \mathbf{\Sigma}^{-1}_{\boldsymbol{v}_r\boldsymbol{v}_r}$ in the remainder of this section.
Initially, an unweighted solution to the homogeneous problem is derived by the use of normal equations. Alternatively, it will be shown under which assumption this problem can be transformed into an inhomogeneous problem to apply ordinary least-squares approaches. Including observation uncertainties, the problem can be extended to a generalized least squares problem. In the latter case, a weighted residual will be minimized using the inverse of the covariance matrix as weighting matrix. Note that the parametrization of the rotation by nine elements in (4.30) is non-minimal. Thus, in the end it will be shown how the final rotation matrix and translation vector can be retrieved from the linear solutions.

Unweighted Solution to the Homogeneous System

In the unweighted case, the normal equations for the problem (4.31) can be derived as:

$$\mathbf{A}^T\mathbf{A}\boldsymbol{u} = \mathbf{N}\boldsymbol{u} = \mathbf{0} \tag{4.33}$$

Following [56, p. 593], a solution to this problem under the constraint $||\boldsymbol{u}||_2 = 1$ can be obtained using the singular value decomposition (SVD) of $\mathbf{N}$.

$$\mathbf{N} = \mathbf{U}\mathbf{D}\mathbf{V}^T \tag{4.34}$$

The solution to the homogeneous system $\boldsymbol{u}^*$ is then given by the right singular vector (columns of $\mathbf{V}$) corresponding to the smallest singular value in $\mathbf{D}$.

Conversion of the Homogeneous Problem

Alternatively, the homogeneous problem can be modified into an inhomogeneous problem as shown in [56, pp. 90–91]. Therefore, a condition on one parameter is imposed. Here, we set $\hat{t}_z = 1$. Then, the resulting inhomogeneous problem can be written as

$$\begin{aligned} \tilde{\mathbf{A}}\tilde{\boldsymbol{u}} &= \tilde{\boldsymbol{b}} \\ \tilde{\mathbf{A}} &= [\mathbf{A}_1, \ldots, \mathbf{A}_{11}] \\ \tilde{\boldsymbol{u}} &= \left[\hat{r}_{11}, \ldots, \hat{t}_y\right]^T \\ \tilde{\boldsymbol{b}} &= -\mathbf{A}_{12} \end{aligned} \tag{4.35}$$

where $\mathbf{A}_i$ denotes the i'th column of $\mathbf{A}$. $\tilde{\mathbf{A}}$ is obtained from truncating $\mathbf{A}$ by the last column, $\tilde{\boldsymbol{u}}$ contains the remaining eleven unconstrained parameters and $\tilde{\boldsymbol{b}}$ is given by the negative last column of $\mathbf{A}$.
This can be treated as an *Ordinary Least-Squares* (OLS) problem of the form

$$\min_{\tilde{\boldsymbol{u}}} \left|\left|\tilde{\mathbf{A}}\tilde{\boldsymbol{u}} - \tilde{\boldsymbol{b}}\right|\right|^2_2 \tag{4.36}$$

For this type of problems, the normal equations can be obtained as [56, p. 591]

$$\tilde{\mathbf{A}}^T \tilde{\mathbf{A}} \tilde{\boldsymbol{u}} = \tilde{\mathbf{A}}^T \tilde{\boldsymbol{b}} \tag{4.37}$$

An algebraic solution is given by

$$\tilde{\boldsymbol{u}}^* = \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{A}}\right)^{-1} \tilde{\mathbf{A}}^T \tilde{\boldsymbol{b}} \tag{4.38}$$

The solution to the original problem is then obtained by $\boldsymbol{u}^* = [\tilde{\boldsymbol{u}}^* \quad 1]^T$.
One major drawback of the described approach is that it fails if the true value of $\hat{t}_z$ is zero or nearly zero. But as in the context of this thesis, the constrained parameter is the distance of ARDEA to the camera in z-direction, the assumption of $\hat{t}_z$ being sufficiently larger than zero is valid.
In many applications, $\tilde{\mathbf{A}}^T \tilde{\mathbf{A}}$ might be poorly conditioned and thus, more stable mehtods are applied, e.g. by using a QR decomposition [55, p. 653]. In the implementation of the approach, the Matlab built-in function `lscov()` is used which is also based on a QR decomposition [57].

Generalized Linear Least-Squares Solution

So far, solutions have been computed only for the unweighted case. In the previous sections, it has been shown that measurement covariances can be propagated into the reduced observation space. The resulting covariance matrix $\boldsymbol{\Sigma}$ implicitly encodes how trustworthy the corresponding observations are. Hence, it is desirable to integrate this information into the problem formulation.
According to [58, pp. 153–186], for the linear and inhomogeneous problem (4.36) with known observation covariances, the best linear unbiased estimate (BLUE) is found solving

$$\min_{\tilde{\boldsymbol{u}}} \left(\tilde{\mathbf{A}}\tilde{\boldsymbol{u}} - \tilde{\boldsymbol{b}}\right)^T \boldsymbol{\Sigma}^{-1} \left(\tilde{\mathbf{A}}\tilde{\boldsymbol{u}} - \tilde{\boldsymbol{b}}\right) \tag{4.39}$$

which is equivalent of the weighted problem formulated in (4.32). This type of problems is commonly also referred to as *Generalized Least-Squares Problem* (GLS).
A common method applied in many GLS problems is using the Cholesky Factorization of the covariance matrix to transform a GLS into an OLS problem. The Cholesky Factorization is given as

$$\boldsymbol{\Sigma}^{-1} = \mathbf{L}\mathbf{L}^T \tag{4.40}$$

with a lower triangular matrix $\mathbf{L}$.
Using this decomposition, (4.39) is equivalent to

$$\min_{\tilde{\boldsymbol{u}}} \left|\left|\mathbf{L}^T \left(\tilde{\mathbf{A}}\tilde{\boldsymbol{u}} - \tilde{\boldsymbol{b}}\right)\right|\right|_2^2 \tag{4.41}$$

Furthermore, together with the transformations $\bar{\mathbf{A}} = \mathbf{L}^T \tilde{\mathbf{A}}$ and $\bar{\boldsymbol{b}} = \mathbf{L}^T \tilde{\boldsymbol{b}}$ we can formulate an equivalent OLS problem as

$$\min_{\tilde{\boldsymbol{u}}} \left|\left|\left(\bar{\mathbf{A}}\tilde{\boldsymbol{u}} - \bar{\boldsymbol{b}}\right)\right|\right|_2^2 \tag{4.42}$$

which can then be solved by the same methods that have been introduced for the ordinary least-squares problem.
It is important to mention here, that calculating the overall covariance matrix $\boldsymbol{\Sigma}$ requires an initial estimate of the orientation. Thus, in practice, first the unweighted problem will be solved once by SVD to compute the covariance matrix for the GLS problem.

Retrieving True Pose

The linear solutions $\boldsymbol{u}^*$ obtained from the various least-squares approaches yield a matrix $\hat{\mathbf{R}}$ and a vector $\hat{\boldsymbol{t}}$ by rearranging the elements:

$$\hat{\mathbf{R}} = \begin{bmatrix} \hat{r}_{11} & \hat{r}_{12} & \hat{r}_{13} \\ \hat{r}_{21} & \hat{r}_{22} & \hat{r}_{23} \\ \hat{r}_{31} & \hat{r}_{32} & \hat{r}_{33} \end{bmatrix} \quad \hat{\boldsymbol{t}} = \begin{bmatrix} \hat{t}_x \\ \hat{t}_y \\ \hat{t}_z \end{bmatrix} \tag{4.43}$$

This determines the unknown pose ${}^c\mathbf{R}_i$, ${}^c\boldsymbol{t}_i$ up to a scale factor. The translational part already points into the right direction but still needs to be scaled appropriately. The required scale factor can be recovered exploting the fact that each column $\hat{\boldsymbol{r}}_1, \hat{\boldsymbol{r}}_2, \hat{\boldsymbol{r}}_3$ of the rotation matrix must have a norm equal to one. This condition results in scaling the translation as follows [8]:

$${}^c\hat{\boldsymbol{t}}_i = \frac{\hat{\boldsymbol{t}}}{\sqrt[3]{||\hat{\boldsymbol{r}}_1||_2 \, ||\hat{\boldsymbol{r}}_2||_2 \, ||\hat{\boldsymbol{r}}_3||_2}} \tag{4.44}$$

In addition to the scale error, the non-minimal representation of the orientation by nine parameters (nine matrix elements) causes $\hat{\mathbf{R}}$ to not define a correct rotation matrix. The estimated orientation ${}^c\hat{\mathbf{R}}_i$ can be recovered from the SVD of $\hat{\mathbf{R}}$

$$\hat{\mathbf{R}} = \mathbf{U}_R \mathbf{D}_R \mathbf{V}_R^T \tag{4.45}$$

The closest possible rotation matrix is then given by [59]

$${}^c\hat{\mathbf{R}}_i = \mathbf{U}_R \begin{bmatrix} 1 & & \\ & 1 & \\ & & \det\left(\mathbf{U}_R \mathbf{V}_R^T\right) \end{bmatrix} \mathbf{V}_R^T \tag{4.46}$$

4.2.3. Non-Linear Optimization

It is very common for many tasks in the field of computer vision to apply a non-linear refinement after obtaining an initial estimate from solving a linear problem. Specifically in this case, we want to apply a non-linear refinement to minimize weighted tangent space residuals from (4.26). The minimization objective function can be formulated as

$$E(\boldsymbol{u}) = \pi_{\text{null}}\left(\hat{\boldsymbol{p}}, \boldsymbol{u}\right)^T \, \boldsymbol{\Sigma}^{-1} \, \pi_{\text{null}}\left(\hat{\boldsymbol{p}}, \boldsymbol{u}\right) \tag{4.47}$$

where $\pi_{\text{null}}\left(\hat{\boldsymbol{p}}, \boldsymbol{u}\right)$ is the stacked vector of nullspace projections of the integrated odometry readings $\{{}^i\hat{\boldsymbol{p}}_j\}_{j=0,\ldots,n}$ using (4.4) and (4.26). A minimal representation of rotations (Rodriguez parametrization [60]) is used for parametrization of $\boldsymbol{u}$ during non-linear optimization.
Equation (4.47) can be rewritten as a common non-linear least-squares problem:

$$E(\boldsymbol{u}) = \left|\left|\mathbf{L}^T \, \pi_{\text{null}}\left(\hat{\boldsymbol{p}}, \boldsymbol{u}\right)\right|\right|_2^2 \tag{4.48}$$

where $\mathbf{L}$ is retrieved from the Cholesky factorization in (4.40).
In this thesis, minimization of (4.48) is done using Matlab's function `lsqnonlin()` which applies a trust-region method to solve the problem. This is why the basic concept of trust-region methods will be revised briefly.

Algorithm 2 Trust-Region Minimization for Non-Linear Least Squares

Problem: $\min_{\boldsymbol{x}} f(\boldsymbol{x})$ with $f(\boldsymbol{x}) = ||\boldsymbol{r}(\boldsymbol{x})||_2^2$
Given: starting point $\boldsymbol{x}_k$
repeat
  Formulate trust-region subproblem by Taylor Series expansion around $\boldsymbol{x}_k$ (equation (4.50))
  Obtain update step $\boldsymbol{s}$ by Gauss-Newton approximation (solve (4.51)
  Accept step if $f(\boldsymbol{x}_k + \boldsymbol{s}) < f(\boldsymbol{x}_k)$
  Adjust size of trust-region Δ_k
until convergence

Trust-Region Method for Non-Linear Least Squares Problems

The first step of trust-region methods is the same as in the well-known Newton method (or the related Gauss-Newton and Levenberg-Marquardt algorithms). For a minimization objective function $f : \mathbb{R}^n \to \mathbb{R}$, a quadratic approximation is obtained in the proximity of a current point $\boldsymbol{x}_k$ using Taylor series expansion. Assuming the objective function to be twice continuously differentiable, this approximation is given by

$$f(\boldsymbol{x}_k + \boldsymbol{s}) \approx f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{s} + \boldsymbol{s}^T \mathbf{H}(\boldsymbol{x}_k) \boldsymbol{s} \tag{4.49}$$

Here, $\nabla f(\boldsymbol{x}_k)$ and $\mathbf{H}(\boldsymbol{x}_k)$ denote the gradient respectively the Hessian matrix at the current point. From now on, the notation will be simplified such that $\mathbf{H}_k = \mathbf{H}(\boldsymbol{x}_k)$ and $\nabla f_k = \nabla f(\boldsymbol{x}_k)$. Trust-region methods then define a region around the current iterate within which they trust this approximation to be a valid representation of the objective function [61, chapter 4]. The update step $\boldsymbol{s}$ is then chosen as the minimizer of the approximation in that region. Thus, the step is a solution to the sub-problem:

$$\min_{\boldsymbol{s} \in \mathbb{R}^n} f_k + \nabla f_k^T \boldsymbol{s} + \boldsymbol{s}^T \mathbf{H}_k \boldsymbol{s} \qquad \text{s.t. } ||\mathbf{D}\boldsymbol{s}||_2 \leqslant \Delta_k \tag{4.50}$$

where $\mathbf{D}$ is a diagonal scaling matrix defining an elliptic trust region with the bound Δ_k.
The step is accepted only if it satisfies $f(\boldsymbol{x}_k + \boldsymbol{s}) < f(\boldsymbol{x}_k)$. Otherwise the size of the trust region will be decreased. Matlab applies standard rules for the adjustment of the trust-region dimension Δ_k. [62]

For the special case of the objective function being a nonlinear least-squares problem, i.e. $f(x) = ||\boldsymbol{r}(\boldsymbol{x})||_2^2$, the solution to (4.50) can be obtained by an approximate Gauss-Newton direction. Therefore the Hessian is approximated using the Jacobian $\mathbf{J}$ of the residual $\boldsymbol{r}$ such that $\mathbf{H} = \mathbf{J}^T\mathbf{J}$. Then the step s is obtained as solution to the normal equations

$$\mathbf{J}^T\mathbf{J}\boldsymbol{s} = -\mathbf{J}^T\boldsymbol{r} \tag{4.51}$$

A very general summary of the trust-region algorithm is given in algorithm 2.

4.3. Overall Algorithm: PnP with Uncertainties

Having derived all steps in detail in the last sections, now a step-by-step description of the overall approach is provided in Algorithm 3. It includes all steps from input processing through spherical normalization, projection into the reduced observation space up to the minimization of a weighted residual in the tangent vector space.

Algorithm 3 PnP with Uncertainties

Input: Camera intrinsics $\mathbf{K}$, 2D image observations $\boldsymbol{x}'_{i-j}$, 2D uncertainties $\boldsymbol{\Sigma}_{\boldsymbol{x}'_{i-j}\boldsymbol{x}'_{i-j}}$, delta poses $\left({}^{i-j-1}\mathbf{R}_{i-j}, {}^{i-j-1}\boldsymbol{t}_{i-j}\right)$, 3D uncertainties $\boldsymbol{\Sigma}_{\delta\boldsymbol{u}_{i-j}\delta\boldsymbol{u}_{i-j}}$

Output: Pose ${}^{c}\mathbf{T}_i = [{}^{c}\mathbf{R}_i \ {}^{c}\boldsymbol{t}_i]$

 # Processing Inputs

for $j = 0$ to n **do**
$\boldsymbol{v}_{i-j} \leftarrow$ Normalized observation using (4.6) and (4.8)
$\mathbf{J}_{\boldsymbol{v}_{i-j}} \leftarrow$ Jacobian element for spherical normalization using (4.12)
$\mathbf{J}_{\boldsymbol{v}_{r,i-j}} \leftarrow$ Nullspace vectors $[\boldsymbol{r}_{i-j} \ \boldsymbol{s}_{i-j}]$ from (4.24)
$\boldsymbol{\Sigma}_{\boldsymbol{v}_{i-j}\boldsymbol{v}_{i-j}} \leftarrow$ Propagate image uncertainties using (4.11) and (4.12)
${}^{i}\hat{\boldsymbol{p}}_j \leftarrow$ Integrate odometry readings ${}^{i}\hat{\boldsymbol{p}}_j$ by (4.5)
$\mathbf{A} \leftarrow$ Compute entries in design matrix using (4.29)
end for

 # Initial Unweighted Estimate

$\mathbf{N} \leftarrow \mathbf{A}^T\mathbf{A}$ Solve homogeneous system (4.33) using SVD (4.34)
${}^{c}\hat{\mathbf{R}}_{i,0}$, ${}^{c}\hat{\boldsymbol{t}}_{i,0} \leftarrow$ Retrieve pose using (4.46) and (4.44)

 # Covariance Propagation

$\mathbf{J}_{\boldsymbol{v}}$, $\mathbf{J}_{\boldsymbol{v}_r}$, ${}^{(2D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}} \leftarrow$ Construct overall Jacobian for spherical normalization (4.21), overall nullspace Jacobian (4.28) and overall propagated image covariances (4.13)
$\mathbf{B} \leftarrow$ Compute using ${}^{c}\hat{\mathbf{R}}_{i,0}$
${}^{(3D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}} \leftarrow$ Overall pose covariance matrix according to (4.20) and (4.22)
$\boldsymbol{\Sigma}_{\boldsymbol{vv}} \leftarrow {}^{(3D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}} + {}^{(2D)}\boldsymbol{\Sigma}_{\boldsymbol{vv}}$
$\boldsymbol{\Sigma}_{\boldsymbol{v}_r\boldsymbol{v}_r} \leftarrow$ Propagate covariances to reduced observation space using (4.27)

 # Weighted Linear Estimate

${}^{c}\hat{\mathbf{R}}_i$, ${}^{c}\hat{\boldsymbol{t}}_i \leftarrow$ GLS Minimization of weighted residual (4.39)
$\mathbf{B}$, $\boldsymbol{\Sigma}_{\boldsymbol{v}_r\boldsymbol{v}_r} \leftarrow$ Update with GLS estimate

 # Non-Linear Refinement

${}^{c}\mathbf{R}_i$, ${}^{c}\boldsymbol{t}_i \leftarrow$ Trust-Region Minimization of (4.48) with initial estimates ${}^{c}\hat{\mathbf{R}}_i$, ${}^{c}\hat{\boldsymbol{t}}_i$

5 Experimental Validation

The goal of this chapter will be to evaluate the tracking approaches from chapter 3 as well as the pose estimation approach from chapter 4 in experimental setups. For that purpose, experimental data has been recorded using ARDEA's and LRU's on-board sensors and the *VICON Motion Capture System*. First, some prerequisites on type and use of data will be provided. Following that, both tracking approaches, conventional and learning-based, will be evaluated on selected data sets in qualitative and quantitative ways. Next, the pose estimation approach will be evaluated in a simulation with artificially created odometry data. Finally, a real-world problem will be presented, solving the overall problem of combined tracking and pose estimation on real data.

5.1. Prerequisites

As prerequisites for an experimental evaluation, brief descriptions of every experimental data set and the circumstances under which each one has been recorded will be provided. Furthermore, it will be presented how data was used for training a RetinaNet detector.

5.1.1. Experimental Data Sets

Data has been collected in indoor and outdoor setups. Images have been recorded using the different cameras of the LRU. Depending on the environment, images have been recorded either from the tele camera or the navigation (color) camera or both. All systems are extrinsically calibrated such that transformations between all frames are known. Furthermore, both of the cameras are calibrated and intrinsics (according to the pinhole projection 4.7) are given by:

$$\begin{aligned} \mathbf{K}_{\text{color}} &= \begin{bmatrix} 1328.89 & 0.066987 & 627.137 \\ 0 & 1328.78 & 486.505 \\ 0 & 0 & 1 \end{bmatrix} \\ \mathbf{K}_{\text{tele}} &= \begin{bmatrix} 21030.9 & -53.248 & 1095.24 \\ 0 & 21005.7 & 923.69 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{5.1}$$

Color camera images are obtained at a frequency of 14 Hz. In contrast, tele camera images have been taken with different exposure times on different data sets due to changing ambient lighting. That is why for some of the outdoor data sets, tele camera images are provided at a frequency of 7 Hz and for others at a frequency of only approximately 1 Hz. While the data sets that have been recorded outdoor are only used for validation of the tracking algorithms, indoor data sets will be also used for a validation of the pose estimation approach. Therefore, in most indoor scenarios, also visual odometry outputs of ARDEA and ground truth positions from the VICON Motion Capture System are recorded. The ground truth positions are only

used for a qualitative evaluation of the pose estimation on real data. The recorded positions are jumping between several points on ARDEA and thus are not reliable enough for a quantitative evaluation. VO readings are obtained at a rate of 7.8 Hz. Due to safety restrictions, ARDEA has been attached to a hinge and has been carried around to create flight trajectories. For all data sets, a brief description of each set will be provided describing the circumstances under which data has been recorded. Indoor experiments are referred to as **Lab XX** whereas outdoor recordings are named **Outdoor XX**.

- **Outdoor 01 - 02:** The person carrying ARDEA is hidden behind a pillar. This leads to a limited range of ARDEA's movement in the image plane. Tele camera images are recorded only at 7 Hz while keeping the camera static.
- **Outdoor 03 - 05:** Color and tele camera images (at a rate of 1 Hz) are recorded. ARDEA is initially located near the camera (good visibility in the color camera image) and is moved around in the proximity of the camera. Later, ARDEA is moving away until it is hardly visible in the color camera image. Cameras are static and the background is varying.
- **Outdoor 06:** Color and tele camera images (at a rate of 1 Hz) are recorded. ARDEA is in an intermediate distance (good visibility in the tele camera image) and moving back and forth while the camera is moving with ARDEA's movement.
- **Outdoor 07:** Color and tele camera images (at a rate of 1 Hz) are recorded. ARDEA is starting close to the camera and is moving very far away up to a final distance of more than 200 metres (**Outdoor 07**).
- **Lab 01:** Only color camera images are recorded. ARDEA is moving around in the lab staying within the view of the static camera.
- **Lab 02 - 04:** Similar to **Lab 01**, but additionally VO readings and ground truth data are recorded.
- **Lab 05:** Similar to **Lab 02 - 04** but with a moving camera.

5.1.2. Training a RetinaNet Detector

This section will briefly describe, how training data for the task of detecting ARDEA is collected from the experimental data sets. After that, important metrics are introduced that are commonly used to evaluate the results of object detection algorithms. Finally, the training process will be evaluated.

Training Data Collection

To avoid expensive manual labeling of training data, a tracker following the conventional approach (chapter 3.2.1) can be tuned on some selected data sets to create labeled training data (images and bounding boxes) on these sample scenarios. This is done for the sequences **Lab 01** and **Outdoor 01 - 05**. As for most scenes, the tele camera images are recorded at very low frame rates ($1Hz$), these are not very well suited for the conventional tracking approach as displacements between subsequent frames become too large to establish correspondence between detections by the presented approach. Thus, only for **Outdoor 01** and **02**, images from the tele camera are used and color camera images otherwise. To achieve larger variation in the training images, not every frame is used to extract training data, but every tenth frame for color camera images and every fifth frame for tele camera images. Dataset augmentation is applied to produce a larger number of training samples with varying shapes and scales.

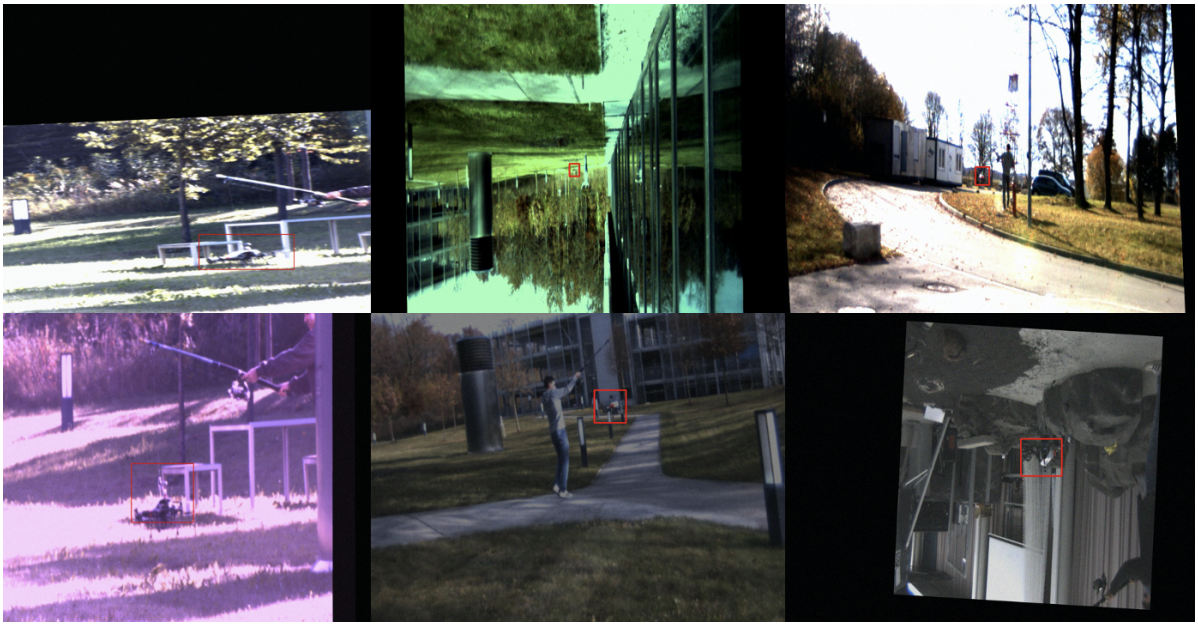


Figure 5.1.: Some examples of augmented training data. Scenes: Outdoor 01 (top left), Outdoor 02 (bottom left), Outdoor 03 (top center), Outdoor 04 (bottom center), Outdoor 05 (top right) and Lab 01 (bottom right).

Operations used for augmentation include random image flips (horizontally and vertically), random crops, Gaussian blur, linear contrast adjustments, Gaussian noise, brightness changes and affine transformations (scaling, translation, rotation, shear). Gaussian noise and brightness changes are sometimes applied channel-wise which might result in changing the color of the image. Some examples of augmented training images with corresponding bounding boxes are shown in Figure 5.1. The resulting augmented data sets are then divided into training and validation data. Bounding boxes extracted for **Outdoor 01**, **04** and **05** are used for training only. **Outdoor 03** is used for validation only. The remaining **Outdoor 02** and **Lab 01** are split (90% training / 10% validation). In total, this procedure results in 1816 images for training and 241 images for validation.

Evaluation Metrics for Object Detection

For validation purposes, the most common metrics are given by the *Mean Average Precision* (mAP) which is based on the definitions of *Precision* and *Recall*. For a better understanding, these quantities are introduced in the following.
Definitions of *Precision* and *Recall* can be given in terms of *True Positives* (TP), *False Positives* (FP), *True Negatives* (TN) and *False Negatives* (FN). By *True Positives*, we denote detections that can be assigned to ground-truth objects. Contrarily, *False Positives* are detections that are not assigned to any ground-truth object. Following the same logic, *False Negatives* are ground-truth objects that are not detected by the network. *True Negatives* is the number of regions that are correctly not assigned to any object. Positive matches between predicted and ground-truth bounding boxes are defined by the pair of boxes with an $IoU \geqslant 0.5$. Using these terms, the *Precision* can be defined as the ratio of correctly assigned detections divided by the total number of detections of our network. Formally, this is given by:

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

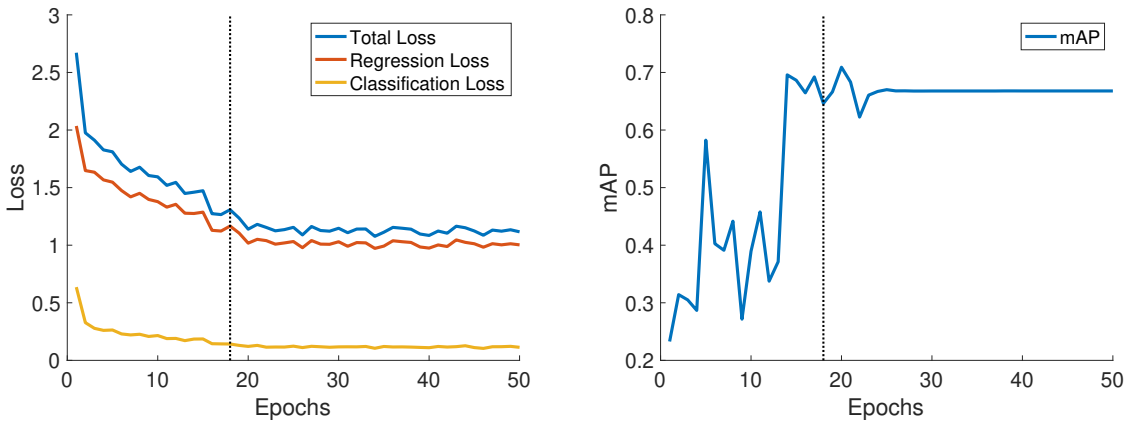
So, the *Precision* gives a quantitative measure for the reliability of the detections but it does not take missing detections into account. Opposed to that, the *Recall* also considers missing detections and is given by the ratio of correctly classified detections divided by the total number of instances of that class. This can be expressed by:

$$Recall = \frac{TP}{TP + FN} \tag{5.3}$$

Based on *Recall* and *Precision*, [63] introduced the *Average Precision* (AP) as evaluation metrics combining the expressiveness of both quantities. The AP is given by the area under the Precision-Recall curve. Usually, the AP would be repeatedly calculated for every class occuring in the object detection task and the results would be averaged to obtain the *Mean Average Precision* (mAP). Note that, in this context only one class of objects is existing and thus, mAP and AP can be used equivalently.

Evaluation of Training Results

Finally, RetinaNet is trained minimizing the total loss as sum of the focal loss for classification (equation (3.14)) and the smooth L1 loss for localization (equation (3.10)). The *Keras* implementation of *RetinaNet* is used and an Adams optimization scheme with an initial learning rate of $1e-04$ is applied. Training is done on batches of size 4 and a total of 50 epochs. The model weights have been pre-trained on the *COCO* benchmark [64]. Loss curves for total loss, box regression, and classification are plotted in Figure 5.2a.
Results on the validation data is monitored by the development of the mAP score which is shown in Figure 5.2b.
These results show that after a number of 14 training epochs, the mAP does not improve significantly for later epochs. Actually, for a very large number of epochs (25 or higher) the mAP is almost constant. This indicates, that training for a large number of epochs does not provide a useful learning information and might even degrade the performance of the network due to overfitting. Moreover, it can be seen that in the range around 20 learning epochs, the loss of the network has nearly converged. The optimal network has been found evaluating snapshots of the detector after 15 to 25 epochs on unseen data. The best results have been achieved by the network after 18 epochs. Hence, the model after training for 18 epochs will be used for tracking purposes.



(a) Loss Curves (b) Validation Results

Figure 5.2.: Loss and Validation Curves (mAP) during training of a RetinaNet Detector for 50 epochs.

5.2. Validation of Tracking Approaches

This section will evaluate both tracking approaches by different quantitative and qualitative means. In the beginning, it will be briefly explained by which measures the performance of each tracking algorithm will be evaluated. Then, each of the tracking approaches will be applied to some of the previously described data sets and both their performances will be assessed.

Validation Metrics

Assuming that the object of interest, which is supposed to be tracked, can only appear once in a scene drastically simplifies the tracking problem. For the learning-based approach, only the best prediction that has a class confidence score larger than 0.5 is considered as a detection of ARDEA. To evaluate the performance of a tracking approach that is simply built upon tracking-by-detection without any mechanism of data association, two things are important to look at. First of all, it is interesting in how many frames, one can actually detect ARDEA. We measure that by the ratio of the number of frames with a detection of ARDEA divided by the total number of frames and will refer to this measure as *Detection Rate*:

$$\text{Detection Rate} = \frac{\#\text{ of frames with detection}}{\#\text{ of total frames}} \tag{5.4}$$

Not only the detection rate is of interest, but also the number of frames in a row in which ARDEA cannot be detected. In fact, not detecting ARDEA in every frame would not harm the goal of estimating the pose. A bigger problem would be, if ARDEA cannot be detected for a large number of subsequent frames. This might degrade the performance of the pose estimation algorithm. Hence, also the length of the longest sequence without detections will be evaluated and denoted as L_{fail}.
Another important issue that often arises with object detectors, especially with learning-based detectors, are *False Positives*. As no mechanism of data association is implemented so far, the detection of a FP would lead to a wrong point in the image plane trajectory. Thus, the number of FPs is also an important indicator to be considered.

As an analogy for the *detection rate*, we define the *Track Rate* for evaluation of the tracking approach based on conventional methods. Therefore we introduce the track length L_{track} which denotes the number of frames for which the object can be successfully tracked. The *Track Rate* is then given by the ratio of the track length and the number of frames in the Object Tracking Stage. Note that initial frames, which are used for learning of the background model, are not considered.

$$\text{Track Rate} = \frac{L_{track}}{\#\text{ of frames in Object Tracking Stage}} \tag{5.5}$$

Moreover, the number of detected objects will be inspected. The conventional tracking approach is not able to identify specifically ARDEA, but will detect all moving objects in a scene. Detecting too many objects limits the computational performance and increases the difficulty of distinguishing between objects.

Tracking Results on Lab Data Sets

In the first part of the evaluation of both tracking approaches, they will be applied to the data sets that have been recorded in the laboratory. Note that the sequence **Lab 01** has partially occured in the training data of the learning-based detector. **Lab 05** contains images of a moving camera. In that case, the conventional tracker will fail and only predictions from the learned RetinaNet Detector are evaluated.

	Lab 01	**Lab 02**	**Lab 03**	**Lab 04**	**Lab 05**
Track Rate [%]	100	99.21	56.21	97.94	X
n_{BG}	180	180	180	180	X
# of Objects	3	3	3	2	X
# of Frames	419	939	1066	957	X

Table 5.1.: Performance indicators for evaluation of the conventional tracking approach.

The conventional tracker from chapter 3 has been implemented for the **Lab** data sets using a number of 180 frames for background learning ($n_{BG} = 180$) and a distance threshold of 30 for the decision whether a pixel is well described by the background model. As a threshold on the IoU for data association between subsequent frames (equation (3.15)), $c_{IoU} = 0.4$ is selected. The resulting tracking performances for indoor experiments are shown in Table 5.1.
The numbers indicate that for indoor experiments, the conventional tracker usually yields very reliable tracking results with track rates of nearly 100% for the sequences **Lab 01, 02** and **04**. That means, after having detected ARDEA, it can be tracked almost until the end of the video. Actually, it just fails because it either lands and remains still (**Lab 02**) or it is occluded by the operator at the end of the sequence (**Lab 04**). Due to very difficult circumstanes, the tracker fails after 56% of the sequence for the set **Lab 03**. The frame during which ARDEA is lost is shown in Figure 5.3. One can observe, that even with the human eye it is hard to distinguish between the back of ARDEA and the background for that specific setup.

Figure 5.3.: Frame of the sequence **Lab 03** where ARDEA's track is lost. Real position indicated by dashed orange box.

Besides ARDEA, the introduced approach for tracking and detection based on a combination of background subtraction and a CFT always detects 1 or 2 additional objects in the scene which correspond to the human operator or the hinge that is used to carry ARDEA.

Now, the RetinaNet detector which has been trained as described in section 5.1.2 is applied to the same data sets. Table 5.2 summarizes the results on these image sequences. It can be observed that for all test sequences, detections of ARDEA without any false positive predictions can be obtained in more than half of the frames. To be more precise, the detection rate ranges from approximately 55% in the worst case (**Lab03**) to more than 80% (**Lab 03**). **Lab 01** is

	Lab 01	**Lab 02**	**Lab 03**	**Lab 04**	**Lab 05**
Detection Rate [%]	100	80.085	63.696	54.859	60.478
L_{fail}	0	44	103	93	25
# of FPs	0	0	0	0	0
# of Frames	419	939	1066	957	856

Table 5.2.: Performance indicators for evaluation of the learning-based tracking-by-detection approach.

also listed here but needs to be treated separately as it was partly included in the training process. Nevertheless, detections are obtained in every frame of the sequence, also the ones that have been excluded from training.
The longest streak without detection varies widely depending on the data set. In the worst case, no detection of ARDEA is obtained for a sequence of almost 10% of the total number of frames. In the best case, this amount reduces to less than 3% (25 out of 856 frames for **Lab 05**). For the conventional tracking approach, one result was that tracking comes especially difficult when the back of ARDEA is facing the camera which is harder to distinguish from the background. This result seems to transfer also to the learning-based case where the longest streaks without detections can be explained by long sequences in the video where only the backside of ARDEA can be seen in the image.
The results for **Lab 05** indicate that for moving camera setups, the learning-based tracking-by-detection approach can be applied without any loss in performance.

For a comparison of the tracking results in terms of the trajectory, Figure 5.4 visualizes the resulting trajectories in the image plane for both approaches. As an exemplary scenario, **Lab 02** is chosen here. It can be seen that the trajectories are very similar but still seem to have an offset. To obtain a quantitative measure by how much the trajectories resulting from the two tracking approaches differ, an average deviation of the 2D image position is computed as

$$\tilde{\boldsymbol{x}}_{\Delta} = \frac{1}{N_M} \sum_{i \in M} ||\tilde{\boldsymbol{x}}_{i,\text{pred}} - \tilde{\boldsymbol{x}}_{i,\text{track}}|| \tag{5.6}$$

where M is the set of frames that contains only frames for which there exists both, a position $\tilde{\boldsymbol{x}}_{i,\text{pred}}$ predicted by the RetinaNet detector, and a position $\tilde{\boldsymbol{x}}_{i,\text{track}}$ obtained by conventional tracking. N_M denotes the number of such frames and is given by $N_M = |M|$. This average deviation is computed for all the indoor data sequences and is listed in Table 5.3. The mean

	Lab 01	**Lab 02**	**Lab 03**	**Lab 04**
$\tilde{\boldsymbol{x}}_{\Delta}$ [pixel]	18.56	13.77	13.58	14.18

Table 5.3.: Average deviation of trajectories in the image plane for Lab Data Sequences with static camera.

absolute distance between the two resulting trajectories is varying between 13 and 19 pixels. Considering raw image dimensions of 1292×964, this represents a deviation of only $1 - 2\%$ with respect to the image size. Furthermore, in the sample data sets, ARDEA is in a close range with mean bounding box dimensions of approximately 113×77 (evaluated for *Lab 02*). The average pixel error per direction for the example is given by 12 pixels (in x) and 5 pixels (in y). Relative to the object's size, this error can be considered quite small.

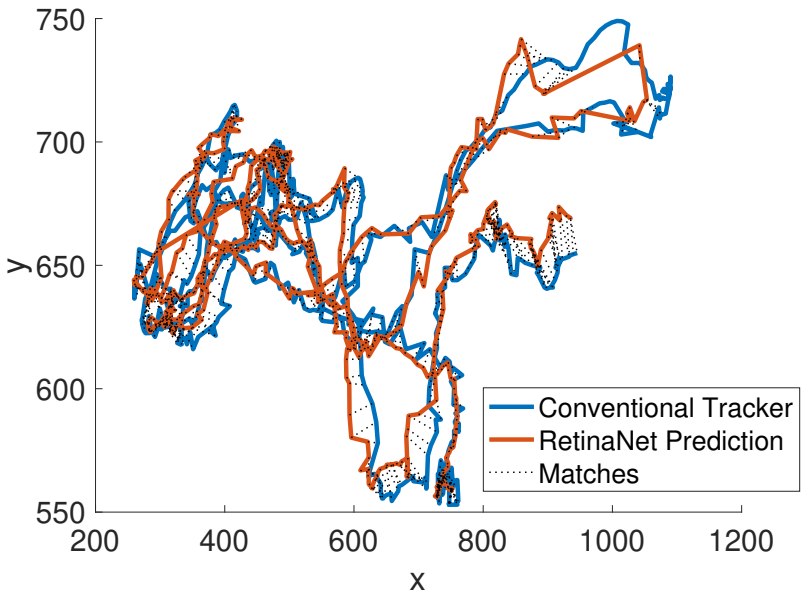


Figure 5.4.: **Lab 02**: Trajectories of ARDEA in the image plane obtained from the two tracking approaches. Positions for matching timestamps are indicated by dotted lines.

Tracking Results on Outdoor Data Sets

Thus far, it has been shown that both presented approaches for detecting respectively tracking ARDEA are able to achieve good results on sample data that has been recorded in a static environment. While in these sample scenarios, ARDEA has been in close distances to the camera, the goal is to be able to track ARDEA also in larger distances and to estimate relative orientations and locations even when ARDEA is far away. For that purpose, most outdoor data sets have been recorded in a way that ARDEA starts close to the LRU and is moving away from it. For these sets, it will be analyzed how long ARDEA can be successfully tracked. This will be evaluated by means of the last frame of the track (respectively the last detection) and the size of the bounding box in that last frame. As the conventional approach is only applicable to the color camera data due to low frequency of the tele camera, this analysis is done on color camera data of the sequences **Outdoor 03 - 05**. All parameters of the conventional tracker are kept the same as in previous experiments. The only difference is that objects of smaller sizes are accepted as candidate regions to be able to track ARDEA in smaller appearances. Table 5.4 summarizes these experiments.

The results prove that the conventional tracker is able to track the object also when it becomes very small in the image frame. The target bounding box, which is closely related to the object dimensions due to background subtraction, decreases down to dimensions of around or even less than 10 pixels in all cases. Opposed to that, in the RetinaNet detector, the target bounding box is limited by the anchor box dimensions and thus will be larger, even if we still detect ARDEA in the same frame. Hence, only for the conventional tracker, the box size can be seen as a measure of the object size in the image.

For these three sequences, the conventional tracker performs at least comparable (**Outdoor 04**) or even better when the object is becoming very small. This might be due to a lack of small representations of the object in the training data. Another reason, which has already been mentioned, is that the choice of anchor boxes and their size limits the detection sizes. These experiments also reveal one big problem of the learning-based detection approach which can be especially observed in the sequence **Outdoor 03**. As Table 5.4 shows, the last true

	Outdoor 03	**Outdoor 04**	**Outdoor 05**
Last Frame of Track (Conventional)	422	372	578
Last True Detection (RetinaNet)	108	412	360
Last Detection(RetinaNet)	1274	412	360
Dimensions (Conventional) [pixels]	8×9	11×9	6×10
Dimensions (RetinaNet) [pixels]	42×31	56×36	43×27
# of Frames	1286	1500	1833

Table 5.4.: Comparison of performance indicators of both tracking approaches for tracked objects becoming small.

detection of ARDEA can be found in frame 108. However, the network still yields detections in later frames. This is a FP match and an example of such a wrong detection can be seen in Figure 5.5b.
One drawback of the conventional method is the lack of identification of the object of interest. As introduced in the theoretical section, all objects being detected are also tracked. We have seen for the indoor environment, that only a small number of objects is detected. In that case, efficiency of the algorithm is not harmed too much. That changes in more complex scenes with more complex illumination conditions. Figure 5.5a illustrates all objects that are detected exemplary for scene **Outdoor 05**. Here, the algorithm detects 23 objects. As one can see in the image, most of them do not correspond to real moving objects but are caused by changing illumination of the sky or surroundings. Most of these objects are lost by the algorithm after a very small number of frames.

Another way to detect ARDEA also in farther distances is to use the tele camera. As tele camera data are mostly only provided at a frame rate of $1Hz$, the conventional tracker cannot be used here. This is why, we will only elaborate the performance of the learning-based detector on two of the data sets. On the one hand, **Outdoor 06** is used with a moving camera setup. On the other hand, in sequence **Outdoor 07**, it will be investigated up to which distances, the

(a) **Outdoor 05** (b) **Outdoor 03**

Figure 5.5.: Drawbacks of both tracking approaches: (a) Conventional Approach: Large number of detected objects (indicated by green bounding boxes) in scenes with changing illumination. (b) RetinaNet Detector: FP Prediction (indicated by red bounding box).

(a) ARDEA in a distance > 100 m (b) FP Detection of human operator

Figure 5.6.: Farthest Detection (a) and exemplary FP detection (b) by RetinaNet Detector on sequence **Outdoor 07**

RetinaNet detector will still find ARDEA in the image.
In the context of a moving camera in sequence **Outdoor 06**, the size of ARDEA in the image will be nearly constant. The detector achieves a detection rate of 63.16% and the longest sequence without detections is 6 frames. But it also detects a number of 2 FPs. It does not make sense to evaluate these numbers also for **Outdoor 07** as with increasing distances and thus decreasing object sizes, at some point the detector will not be able to find the object in the image plane. There, the last true detection of ARDEA is obtained in frame 181 (out of 247 total frames). Figure 5.6a shows the last detection. During the image sequence the distance has increased up to more than 200 metres (measured using *Google Maps*). Thus, using the tele camera, reliable RetinaNet predictions for ARDEA's positions could be obtained for distances of more than 100 metres. However, especially in latter frames, the learning-based approach again suffers from detection of FPs (Figure 5.6b).

Concluding Remarks on Tracking Approaches

This section evaluated the performance of both tracking approaches that had been introduced in chapter 3. In first experiments, it has been shown that both perform quite well in an indoor and static environment with the object of interest (ARDEA) being in a close range to the camera. Nevertheless, both still have some significant drawbacks when applied to more complex setups. The conventional approach mainly suffers from the lack of object identification and thus has to track a large number of objects that are wrongly detected as moving objects due to illumination changes in the scene. In contrast, the RetinaNet based tracking-by-detection approach seems to have better generalization capabilities but its performance degrades due to FP predictions. It must be mentioned here, that the focus of this thesis has not been on the training of the network and thus, the training process itself has a big potential for improvement by e.g. gathering more diverse data.
However, it needs to be stated that with these two approaches, reliable tracking could be achieved on simple scenarios and tracking respectively detection of ARDEA in very large distances is enabled.

5.3. Validation of Pose Estimation Approach

The goal of this section is to validate the novel approach for pose estimation including full uncertainty information that was introduced in chapter 4. First, validation metrics are defined to evaluate the performance of the new method compared to other state-of-the-art approaches. Following that, the approach will be applied in a simulation and its performance will be evaluated in various settings. In the end, the method will also be applied in a real experiment with real-world data.

5.3.1. Validation Metrics

The accuracy of the estimated pose of ARDEA with respect to the camera frame is evaluated separately for translation and orientation. The translational error ϵ_t for an estimated translation vector $\boldsymbol{t}_{\text{est}}$ and a ground truth translation vector $\boldsymbol{t}_{\text{gt}}$ is given by the relative deviation.

$$\epsilon_t = \frac{||\boldsymbol{t}_{\text{gt}} - \boldsymbol{t}_{\text{est}}||_2}{||\boldsymbol{t}_{\text{gt}}||_2} \tag{5.7}$$

The rotational error ϵ_r in degree is measured as the angle of the delta orientation $\mathbf{R}_\Delta$ between the ground-truth orientation $\mathbf{R}_{\text{gt}}$ and the estimated orientation $\mathbf{R}_{\text{est}}$. The angle can be obtained using the axis-angle representation of a rotation matrix.

$$\epsilon_r = angle\left(\mathbf{R}_\Delta\right) = angle\left(\mathbf{R}_{\text{gt}}^{-1}\mathbf{R}_{\text{est}}\right) \tag{5.8}$$

The angle θ of a rotation matrix $\mathbf{R}$ can be calculated using its trace and Rodriguez' Formula [60] as follows:

$$\theta = \arccos\left(\frac{tr\left(\mathbf{R}\right) - 1}{2}\right) \tag{5.9}$$

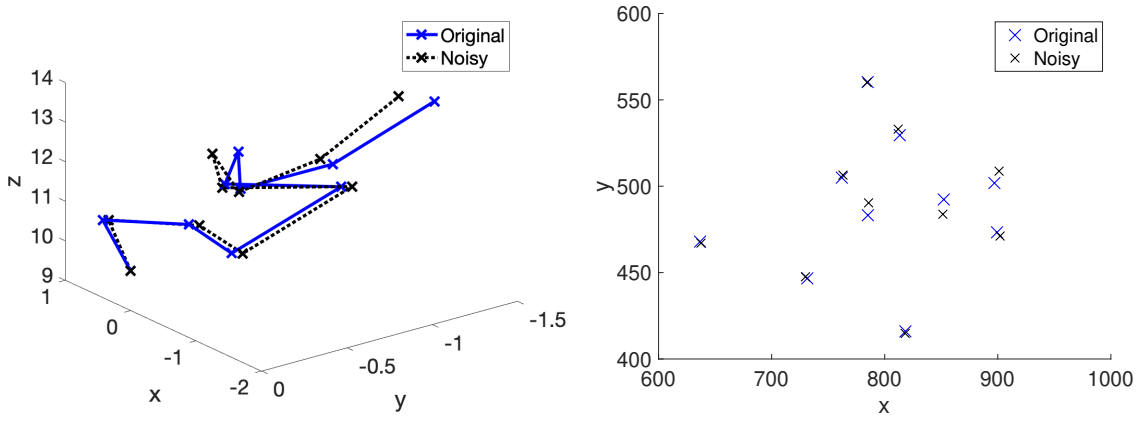
In this thesis, all results will be compared to MLPnP and its publicly available implementation. In the original paper [8], an extensive review of state-of-the-art PnP solvers and an evaluation of their performances compared to MLPnP can be found. The authors have shown MLPnP's great performance in presence of 2D observation uncertainties, outperforming most other methods in terms of accuracy as well as runtime. Thus, only MLPnP will be used as a benchmark here and other solvers are not considered.

5.3.2. Simulation

For a simulation of the approach, it is necessary to create artificial trajectories and the corresponding odometry readings and image points. Furthermore, these data have to be disturbed by a random noise and the corresponding uncertainties need to be modeled. The procedure of data creation will be described in the following.

Trajectory Generation

An arbitrary starting point of the trajectory in the camera frame can be set (${}^c\boldsymbol{p}_{i-n}$ with the notation of chapter 4). Starting from that initial point of the trajectory, a number of n steps is obtained from a random uniform distribution on the unit sphere. These steps of unit length are then concatenated to yield the points ${}^c\boldsymbol{p}_{i-j}$ of the trajectory in the camera frame. Odometry readings ${}^{i-j-1}\mathbf{R}_{i-j}$ and ${}^{i-j-1}\boldsymbol{t}_{i-j}$ (delta orientations and translation) between subsequent frames are computed assuming random relative orientations. Image points corresponding to the artificial trajectories can be obtained by a simple pinhole projection assuming a known camera intrinsics matrix $\mathbf{K}$.



(a) Disturbed trajectory with $k = 3$ (b) Disturbed image observations with $\sigma_{2D,max} = 5$

Figure 5.7.: Example scenario for an artificially created trajectory and corresponding 2D observations including noises ($n = 10$).

Noise Generation

Now, these artificially generated odometry readings and image observations are corrupted by random noise. For the 2D observations, a maximum standard deviation $\sigma_{2D,max}$ (in pixel units) is defined. Within the range of this parameter, for every observation a random factor is drawn as standard deviation of a Gaussian noise for that observation. For the following experiments, this parameter is set to $\sigma_{2D,max} = 5$ pixels. The odometry readings are also disturbed by Gaussian noise with the following standard deviations

$$\begin{aligned} \sigma_t &= k \begin{bmatrix} 1 \times 10^{-3} & 5 \times 10^{-3} & 9 \times 10^{-3} \end{bmatrix} \\ \sigma_r &= k \begin{bmatrix} 1 \times 10^{-2} & 2 \times 10^{-2} & 2 \times 10^{-2} \end{bmatrix} \end{aligned} \tag{5.10}$$

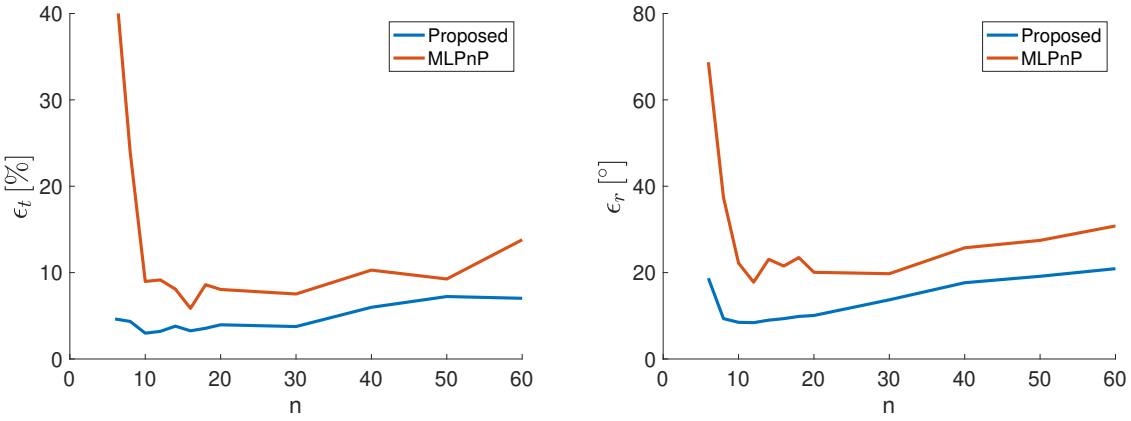
where k denotes a factor, that will be used to scale the noise in some of the experiments. Standard deviations are given in meters respectively degrees. The numbers in (5.10) are not randomly chosen but have been obtained by averaging the errors in a real-world visual odometry application. In that case, the step size was not necessarily equal to 1 which is why the scale factor is used here to simulate different conditions.

As an example, Figure 5.7 illustrates an exemplary scenario of the simulation including the original trajectory, image observations and the perturbed data. The drift of the noisy trajectory with regard to trajectory length in this case is approximately 3%.

Results

Validation is executed based on a Monte Carlo Simulation. Therefore, the described procedure for data generation is repeated for 10 different samplings of the 3D points and each of the trajectories as well as corresponding image observations are perturbed by 50 different Gaussian noises of the same standard deviation. For each of the 500 resulting setups, the modified PnP problem is solved once using the original MLPnP and once using the newly proposed method. The mean of the resulting pose accuracies is computed to compare the two algorithms. Simulating several different noises and samplings makes the simulation more robust in a statistical sense.

As a first experiment, the influence of the number of points of the trajectory is investigated. For that purpose, the noise scale factor is set to a fixed value $k = 3$. Then, the number of points n of the trajectory is varied in a range of 6 (minimum number of required correspondences to obtain an initial solution) to 60 points leading to an average drift of $3 - 5\%$. The results are



(a) Mean Translational Error

(b) Mean Rotational Error

Figure 5.8.: Comparison of mean pose errors for varying number of trajectory points n.

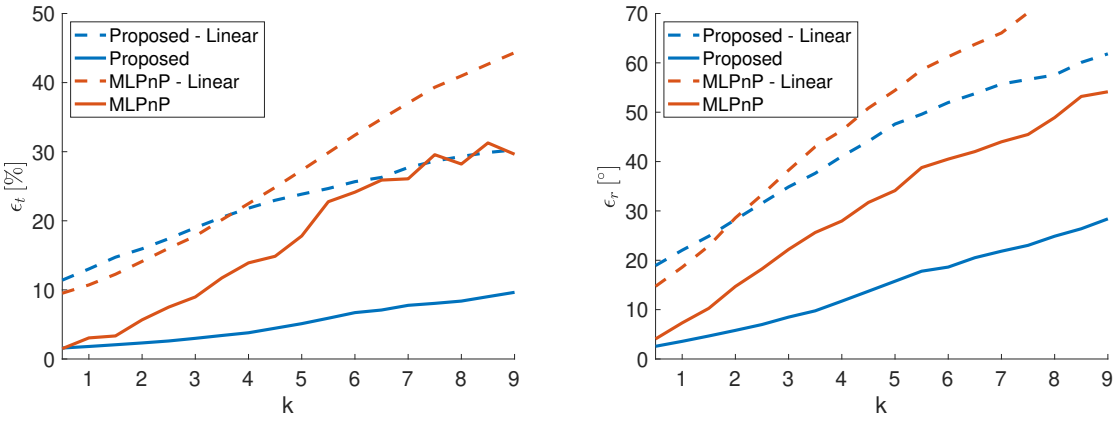
shown in Figure 5.8.

It can be observed that with the proposed new method and for this specific experiment, a minimum pose error seems to be obtained with a number of only 10 points. Adding more points does not yield a significant improvement of the pose errors. Instead, results might even degrade when adding a larger number of points. This result is to some degree expected, as due to the sampling strategy, an increase in the number of points comes along with longer trajectories. As subsequent odometry readings are not independent and errors will sum up, longer trajectories will also have larger errors for points being farther in the past. Hence, the additional points do not provide any useful information.

The plots clearly show the benefit of integrating 3D covariance information into pose estimation. The proposed method constantly outperforms the MLPnP approach in terms of translational and rotational accuracy by a large margin. For estimates of the translation, the error is reduced by $3-5\%$ in absolute percentages. For orientation estimates, the error is reduced by $5-10$ degrees. To emphasize the benefit of the proposed approach, that means a reduction by up to 50% for both, the translational and rotational error.

In a second experiment, the number of points is kept constant ($n = 10$) and the noise scale factor k will be varied. Varying this factor leads to larger errors of the absolute 3D points while keeping the trajectory length almost constant. The results of this investigation are presented in Figure 5.9. It shows the mean pose errors for a variation of the noise factor k between $k = 0.5$ to 9 causing an average drift of the trajectories from 0.5% up to almost 10%. This time, also the results before non-linear optimization are depicted. In the case of the original MLPnP, the linear solution is obtained from SVD whereas in the proposed method, it is obtained from solving a GLS problem. For the linear solution, it can be observed that it can be improved especially for increasing noise magnitudes. If the noise level of the odometry readings is very low, the GLS solution might even be slightly worse. For large noise levels, the GLS solution even yields results of the same quality as MLPnP after non-linear refinement.

The full potential of the newly proposed method is demonstrated by the results after non-linear optimization. It is able to reduce the translational and the rotational errors significantly across all noise levels. Moreover, it can be observed that the gap between the two approaches increases with increasing noise level. For the maximum noise level ($k = 9$) with a trajectory drift of 10%, the error in the position can be reduced from 30% to just below 10%. The error of the relative orientation drops from more than 50 degrees to 28 degrees.

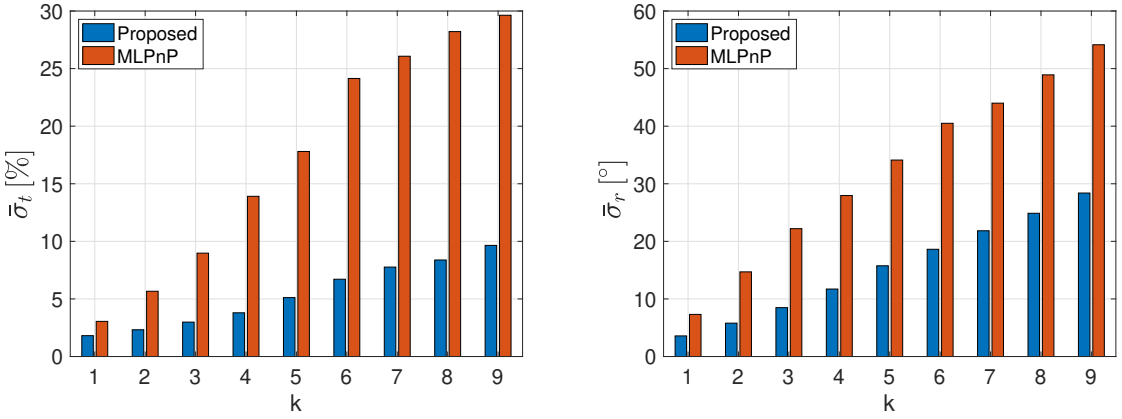


(a) Mean Translational Error (b) Mean Rotational Error

Figure 5.9.: Comparison of mean pose errors for varying noise levels k.

It has been shown, that integrating 3D covariance information into a PnP problem can significantly improve solution quality. The results have been evaluated in a statistical sense by averaging over a number of 500 simulations. In addition to showing the average pose errors, it might be of interest to examine the robustness of the algorithm. For that purpose, not only means of the resulting pose errors are computed, but also the corresponding error standard deviations $\bar{\sigma}_t$ and $\bar{\sigma}_r$. These standard deviations are depicted in Figure 5.10. It clearly shows that, not only the resulting mean pose estimates are improved but also the standard deviation of the pose results is remarkably lower. Thus, the new method for pose estimation including 3D covariances not only yields better results on average, but it also seems more robust to noise.

One outstanding advantage of MLPnP was its real-time capability with an enormous efficiency which has been able to compete with EPnP, the fastest state-of-the-art PnP solver. In Table 5.5, runtimes are compared during the experiment with varying number of points. The simulations are implemented in Matlab and conducted on a Laptop with Intel Dual Core i5@3,1 GHz. The numbers show that the boost in accuracy comes with a significant loss in efficiency with



(a) Standard deviation of the translational error (b) Standard deviation of the rotational error

Figure 5.10.: Comparison of standard deviations of the pose errors for varying noise level.

MLPnP being a lot faster. One major reason is the more expensive optimization that is run in the newly proposed method. However, with maximum execution times of approximately 36 milliseconds, the new algorithm is still able to operate at more than 20 frames per second which is still nearly real-time. In the region of $10-20$ points, where also the minimum pose error was found in the simulation, the execution time is only $10-12$ miliseconds. Furthermore, there is a potential for improvement concerning efficiency as Jacobians are computed numerically in the current implementation.

n	6	8	10	12	14	16	18	20	30	40	50	60
Proposed	18.1	10.7	12.2	10.4	10.9	11.5	12.0	12.0	17.2	21.4	27.5	35.8
MLPnP	2.2	1.7	2.0	1.9	2.0	2.1	2.2	2.3	3.0	3.5	4.3	4.9

Table 5.5.: Runtime comparison of the two algorithms (average runtimes in milliseconds).

5.3.3. Real-World Experiment

Now, the proposed approach for pose estimation is supposed to be also validated in a real-world experimental setup. As a qualitative measure of accuracy, resulting pose estimates are compared to ground truth positions of ARDEA recorded for the image sequence **Lab 02**.

Ground Truth Measurements

Ground truth (GT) measurements are provided as the positions and orientations of the camera frame of ARDEA with respect to the frame of the LRU color camera. These ground truth positions are unfortunately not very precisely determined as the measurements of the position of ARDEA has been jumping between several markers on ARDEA. This can be seen in Figure 5.11 which exemplary illustrates the measured positions during the sequence **Lab 02** split into its components for every axis direction. Nevertheless, the provided positions still can be assumed somewhere on ARDEA such that pose estimates for the position still should be in a close range to these recorded positions.

Visual Odometry Readings

Delta poses and orientations per time step are provided from ARDEA's VO outputs (*VO_01*). Before solving the modified PnP problem defined in chapter (4), odometry readings are always reduced by integrating up a sequence of odometry readings to yield a fewer number of larger

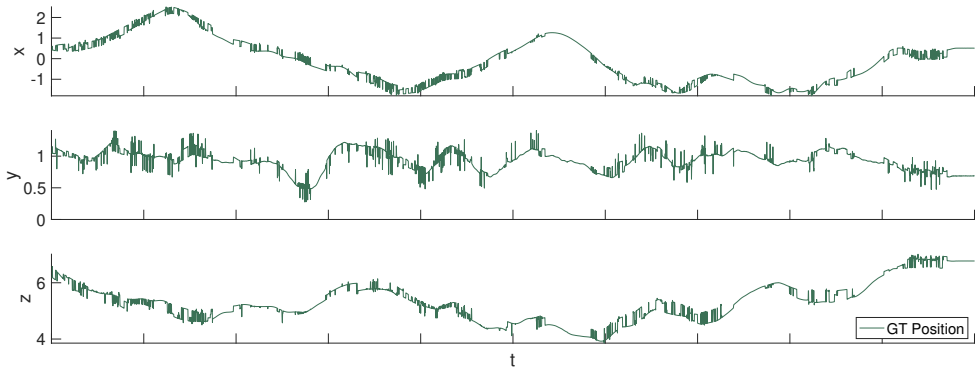


Figure 5.11.: Measured position of ARDEA's camera frame in the frame of the LRU color camera split into x- y- and z-component (**Lab 02**).

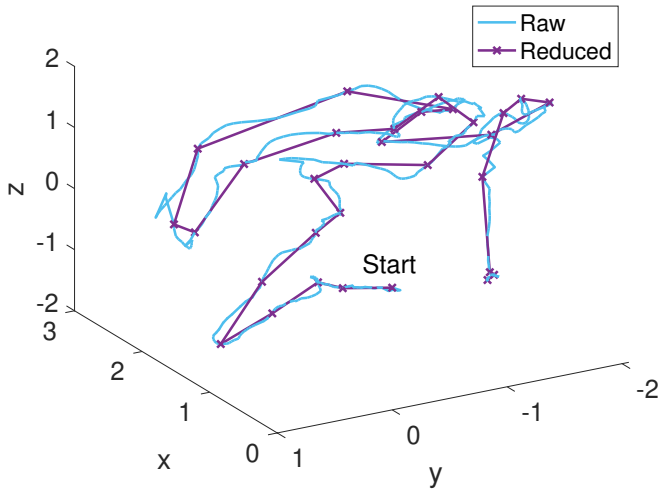


Figure 5.12.: Raw Odometry Readings are reduced to a lower number of readings yielding a reduced trajectory.

VO steps. This is shown in Figure 5.12 where the overall odometry readings are reduced from $n = 520$ to a number of $n_{\text{red}} = 30$.

2D Observations

As 2D observations corresponding to the odometry readings, tracking results are obtained using the conventional tracking approach presented in chapter 3.2.1. The conventional approach is chosen over the learning-based approach here because of its excellent tracking rate on the **Lab** data sets. Recall, that the average deviation $\tilde{\boldsymbol{x}}_{\Delta}$ between the 2D tracking position obtained from conventional and learning-based tracking was roughly 14 pixels (see Table 5.3). We use this as an assumption for the tracking uncertainty with $\sigma_{x'} = \sigma_{y'} = 14$.

Estimating Pose Covariance

In many applications it is desirable to have a measure of reliability about the estimated pose. The standard deviation of the estimated pose parameters can be such a measure for intstance. With the Jacobian $\mathbf{J}$ of the residual equation (4.48) and the result of the minimization $\boldsymbol{u} = [\hat{\mathbf{r}}, \hat{\mathbf{t}}] \in \mathbb{R}^6$, the covariance matrix of the estimated rotation and translation parameters can be obtained as:

$$\boldsymbol{\Sigma}_{\hat{\mathbf{r}}\hat{\mathbf{t}}} = \left(\mathbf{J}^T\mathbf{J}\right)^{-1} \tag{5.11}$$

The vector of standard deviations of the estimated pose can be computed using:

$$\boldsymbol{\sigma}_{\hat{\mathbf{r}}\hat{\mathbf{t}}} = \sqrt{diag\left(\boldsymbol{\Sigma}_{\hat{\mathbf{r}}\hat{\mathbf{t}}}\right)} \tag{5.12}$$

Evaluation

Having gathered all components needed for the pose estimation approach, we can use it to estimate the pose on the recorded data. We want to solve for the pose at different points in time to compare an estimated trajectory qualitatively to a measured GT trajectory. Therefore, always an interval of 150 odometry readings is reduced to 12 delta poses and orientations as well as the corresponding uncertainties. Intervals obtained in that way have an average traveling distance of 6.5 metres. The corresponding 2D observations are obtained by a basic nearest neighbor search of the end timestamps of the odometry sequences and the timestamps of tracking observations. To avoid intervals of very small movements of ARDEA a threshold

on the distance travelled during the interval is imposed. Then, finally the pose at the end of each interval is estimated and compared to the GT position. Additionally, pose uncertainties following equations (5.11)-(5.12) are computed to check if results with a bad accuracy can be recognized by high pose uncertainties. Figures 5.13-5.15 show the resulting estimates for the translation components $\hat{t}_x$, $\hat{t}_y$ and $\hat{t}_z$ of the estimated pose with corresponding estimated uncertainties $\sigma_{\mathrm{t},x}$, $\sigma_{\mathrm{t},y}$ and $\sigma_{\mathrm{t},z}$. Estimates of the translational components are compared qualitatively to the ground truth trajectory ($t_{x,gt}$, $t_{y,gt}$ and $t_{z,gt}$) as well as non-weighted estimates ($\hat{t}_{x,nw}$, $\hat{t}_{y,nw}$ and $\hat{t}_{z,nw}$).
It can be observed that the proposed weighted pose estimation seems to be more stable than the unweighted version. The unweighted version has very sharp peaks which often correspond to sign changes which might be caused from different local minima in the optimization. Comparing the weighted solutions visually to the ground-truth trajectory, it also seems to be more accurate than the unweighted version. While positions in x-direction can estimated very accurately, the positions in y-direction and especially in z-direction seem to be worse with larger deviations and noises in the estimated trajectory components. This result also follows from the fact that the recorded data have larger movements in x than in the other directions and can thus be estimated more accurately.
Another beneficial conclusion from these plots is that in time range where the pose estimates seem to have larger oscillations and errors also the estimated standard deviation of the corresponding pose parameter is higher. As an example, in the range of t_i = 150 - 200, the estimated $\hat{t}_x$ differs from the ground-truth trajectory by a larger margin than otherwise. This larger error can then be also observed in the larger value for $\sigma_{\hat{t},x}$. Similar trends can be seen for all three directions. Moreover, also the value of the estimated standard deviations are in the range of the absolute trajectory error.

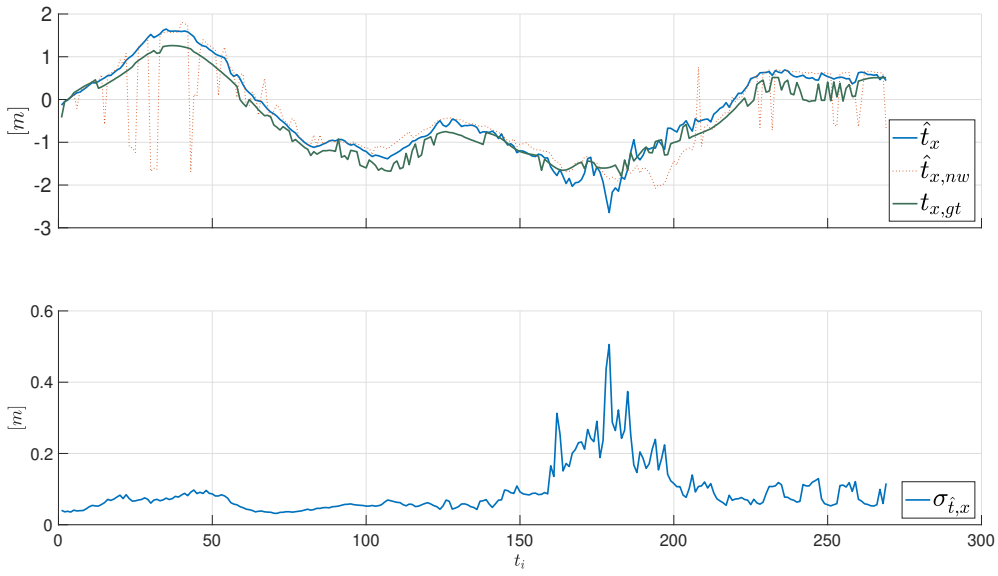


Figure 5.13.: Translational component $\hat{t}_x$ of estimated pose and corresponding estimated standard deviation $\sigma_{\hat{t},x}$ for pose estimation on **Lab02**.

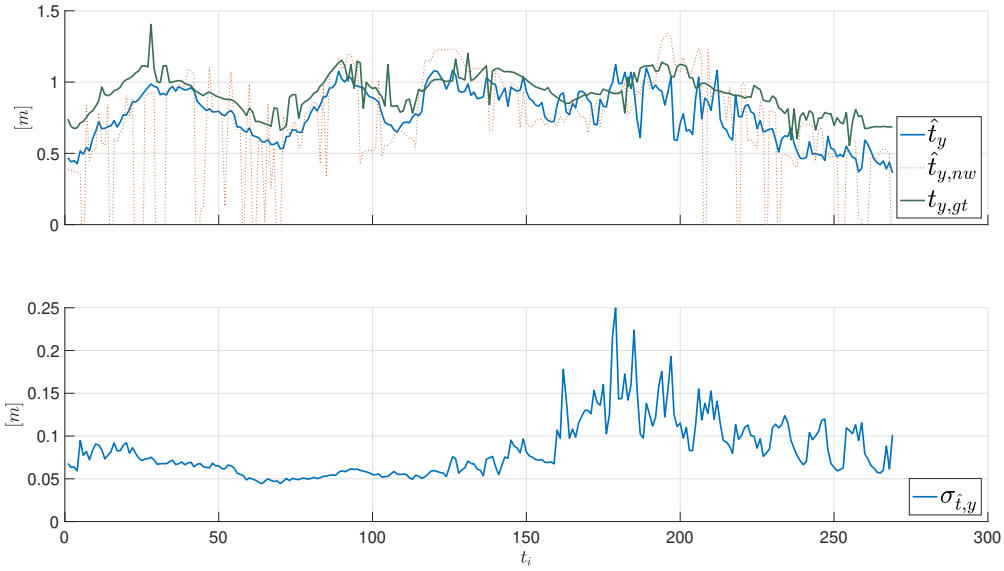


Figure 5.14.: Translational component $\hat{t}_y$ of estimated pose and corresponding estimated standard deviation $\sigma_{\hat{t},y}$ for pose estimation on **Lab02**.

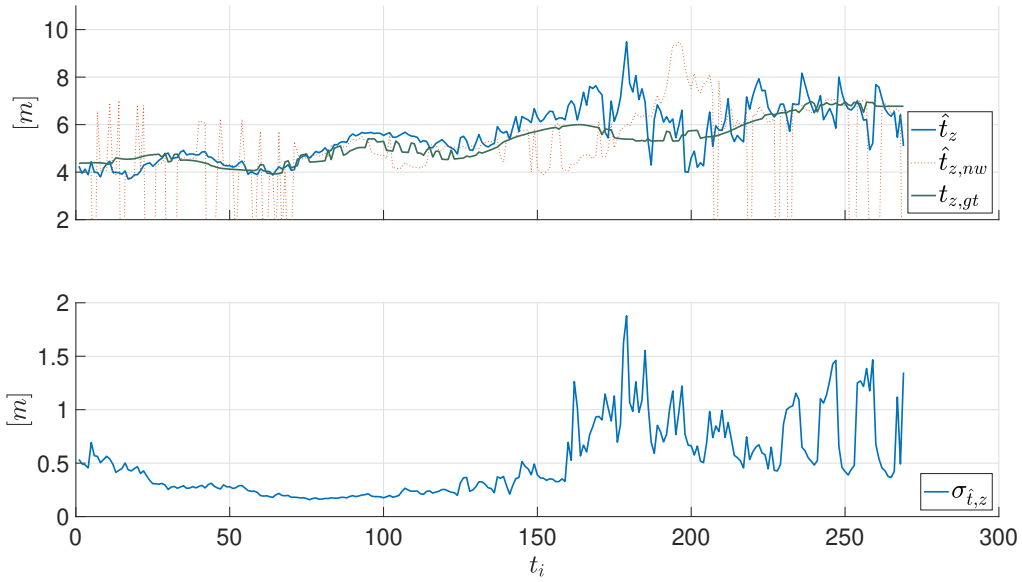


Figure 5.15.: Translational component $\hat{t}_z$ of estimated pose and corresponding estimated standard deviation $\sigma_{\hat{t},z}$ for pose estimation on **Lab02**.

6 Conclusion

This chapter concludes the thesis by summarizing the key achievements of this work and discusses some important aspects of the previous chapters. In the end, a brief outlook will give some ideas on potential future research directions for the covered topics.

6.1. Summary

As described in the introduction, the goal of this thesis has been to estimate the relative position and orientation between a flying robot (ARDEA) and a rover system (LRU) based on visual odometry outputs of ARDEA and images recorded by the LRU. The problem has been divided into the two sub-problems of object detection and tracking respectively pose estimation which have been treated separately in a first step.

Detection and Tracking of ARDEA

For the goal of detecting and tracking ARDEA in an image sequence, two different approaches have been introduced and compared in different environments. On the one hand, an approach based on conventional methods has been introduced which combines a background subtraction algorithm together with a correlation filter based tracker. Assuming a nearly static background scene with constant illumination and a low number of moving objects in a scene, the presented approach has been able to achieve very reliable tracking. Furthermore, it has been shown that it still reliably tracks ARDEA even if the object is becoming very small. However, when applied to more complex environments, the approach suffered from detecting too many objects which often are caused by changing illumination. As no mechanism of identifying the object of interest has been implemented, a very large number of objects has to be tracked which significantly limits computational performance.

As an alternative, a different approach to tracking-by-detection has been proposed based on a learned RetinaNet Detector. Therefore a pre-trained network has been re-trained on detecting only a single class, namely ARDEA. Training data has been extracted from sample scenes by applying the previous conventional tracker and extracting bounding boxes corresponding to ARDEA. Additionally, data augmentation has been applied to create training data with a higher variance in the target's shape and appearance. A small number of epochs has been sufficient to achieve convergence. The trained detector was then evaluated on unseen data. Under the assumption of the target object appearing only once in the scene, only the detection with maximum confidence score above a threshold (typically 0.5) is taken as predicted location of ARDEA. Similarly to the conventional approach, it achieved good detection performance (up to 80% detection rate) on data in simple scenarios where ARDEA is in a close range to the camera and thus quite large in the image. In that setup, hardly any false positives are detected. That changed when applied to more complex data and ARDEA moving farther away. The size

of detected objects is to some degree limited by the size of predefined anchor boxes and thus, the ability to detect ARDEA in far distances is limited. For these complex scenes, also a larger number of false positives has been detected. The distance within which ARDEA is detected can be extended by using tele camera images. Using the tele camera, ARDEA can be detected in distances widely above one hundred metres.

Pose Estimation

The task of pose estimation has been formulated as a temporal version of a PnP problem where the pose of the camera is estimated based on 3D-2D correspondences given as points of a trajectory over time. Furthermore, the PnP formulation has been rewritten in a way that delta poses and orientations from a VO are used as 3D observations instead of absolute 3D points. Additionally, VO uncertainties are leveraged to improve the pose estimation. All given quantities and uncertainties are propagated into a reduced observation space given by tangential planes to the unit sphere in the camera frame. A generalized least squares problem is derived to obtain an initial linear initial estimate of the camera pose which is then refined in a non-linear optimization scheme minimizing a weighted residual in the reduced observation space. Including model uncertainties in the 2D as well as 3D observations allowed to improve the accuracy of the estimated pose, both for the translational as well as the rotational accuracy. In a Monte-Carlo simulation, artificial trajectories have been created with various numbers of points and different Gaussian noises have been applied to perturb the trajectories. In the described setup, the proposed new algorithm outperformed the existing MLPnP framework by far in terms of accuracy and robustness. The only drawback compared to MLPnP is its lower computational efficiency.

Combined Real-World Experiment

In the end, the new approach has been tested on real-world data. Therefore, real VO outputs of ARDEA have been recorded together with the corresponding image frames. Ground truth data were only available as a qualitative reference. 2D observations have been obtained using the conventional tracking approach.
It could be demonstrated that the proposed new approach for pose estimation yielded more stable and accurate estimates of ARDEA's position in the camera frame compared to an unweighted estimation. Moreover, covariance estimates have been derived for the resulting pose parameters. As a measure of reliability they enabled the identification of estimates which are obtained at a high degree of uncertainty.

Discussion

The majority of issues that emerged during this work, and especially during experimental evaluation, are in some way related data quality respectively the way in which data was recorded. Erroneous ground truth measurements prohibited a quantitative evaluation of the presented approaches for tracking and the proposed pose estimation algorithm.
Moreover, the setup of ARDEA being attached to the end of a hinge which has to be carried around to simulate a flight trajectory involved several drawbacks. Especially for the task of tracking, this setup increases the level of difficulty. The ongoing presence of a human operator affects both of the aforementioned tracking approaches. For the conventional tracker, there will always be an additional number of objects (hinge, human) that will be detected as a foreground object and need to be tracked. In contrast, the performance of the learning-based detector might be harmed by learning the presence of a human near ARDEA as a feature of ARDEA. Furthermore, attaching ARDEA to a hinge also results in a couple of problems. First of all, it is hard to control the yaw angle of ARDEA which leads to potentially large and fast rotations around the yaw axis. Next, that resulted in a lot of frames where only the back of ARDEA was visible in the image which is especially for the lab sequences hard to distinguish from the

Figure 6.1.: Prediction Output of RetinaNet Detector for a distant view of ARDEA for the far range image from chapter 1.

background. Last, the range of movement was physically restricted by the range of the hinge. Altogether, with the recorded experimental data sets we have been facing challenges that might not arise in real scenarios.
Nevertheless, despite all these issues, we were still able to achieve quite reliable tracking for sample sets and have been able to prove the potential of the proposed uncertainty-aware method for pose estimation based on tracking and VO results. Besides, another great benefit of the new approach is that it provides an uncertainty measure of the estimated pose to identify potentially bad results.

Finally, let us step back to the example (Figure 1.3) in the introduction where we observed that a reliable feature matching was almost intractable. That was the reason why we solved a temporal PnP problem using correspondences over time instead of 3D-2D keypoint correspondences in a single frame. It has been stated that ARDEA in the far range image had an approximate size of 50×50 pixels. The results for validation of the tracking approach showed that, in general, tracking was able down to sizes of ARDEA below 10 pixels. Also, on the very same image, the trained RetinaNet Detector is still able to predict the correct position of ARDEA (see Figure 6.1).

6.2. Future Work

Due to the high complexity of the problem including a large variety of influential factors, there is potential for improvement in a lot of areas.
Especially when it comes to detection and tracking of ARDEA, we have seen that the presented approaches are able to achieve accurate tracking in simple scenarios but had their limitations in more complex environments. It has been demonstrated how the conventional approach supported the learning-based approach by generating labeled training data. But not only by

generating training data the two could benefit of each other. A combination of both approaches might overcome some of the challenges, both of the individual tracking approaches were facing. In particular, on the one hand, the RetinaNet detector could help to identify objects being detected by the conventional approach. On the other hand, the conventional approach could overcome frame sequences for which the learning-based detector had a long streak of non-detections.
Moreover, also the learning-based detector itself has a big potential for improvement. First of all, training the detector should be enhanced by gathering more diverse data with varying appearances of ARDEA in various environment. This might lead to a boost in performance.
During prediction phase, one could focus on previous locations of ARDEA. As it is safe to assume that the movement of ARDEA is fairly small between subsequent frames (depending on the frame rate), ARDEA should be found in the proximity of the previous location. This would add a data association process for tracking such that the number of false positives might be decreased. Additionally it might reduce the loss of information which is caused by downscaling the images to the default RetinaNet input size.
Concerning the pose estimation, an interesting topic for future research is the selection of observations. During this work, 3D observations have been chosen by a very simple strategy condensing odometry outputs of fixed time intervals to a reduced number of observations thresholded by a minimum traveled distance. Corresponding 2D observations are then obtained by nearest neighbor time step matching. However, there are probably better ways to select the observations in a way to find the optimal pose estimate. Another idea towards the same issue might be to actively generate flight trajectories of ARDEA such that the pose can be estimated as accurately as possible.
All of these aforementioned ideas arose throughout the course of this thesis. Due to limited time we have not been able to realize them although they seem very promising and should be kept in mind for follow-up research.

List of Figures

1.1. Sketch of a planetary exploration mission using a heterogeneous team of robots [3]. 1
1.2. DLR's mobile robots LRU1 (bottom left), LRU2 (bottom right) and ARDEA (top) during experiments at a Moon-analagoue test site on Mt. Etna, Italy [4]. 2
1.3. Region of ARDEA is extracted for close range image (bottom left) and far range image (top left). ORB features are computed using the *OpenCV* implementation. Best four matches are highlighted. 3

3.1. Working Principle of Object Detection using Background Subtraction. 10
3.2. Schematic Overview of a CFT approach [51] 11
3.3. Network architecture of RetinaNet consisting of (b) a Feature Pyramid Network (FPN) on top of a (a) ResNet architecture as backbone with two task-specific subnetworks for (c) classification and (d) bounding box regression [33]. 13
3.4. Visualization of object detections for a given foreground mask resulting from background subtraction 15
3.5. Visualization of the tracking target states. *TO*s are indicated by green bounding boxes, *COR*s by blue bounding boxes. Left: Both *TO*s representing foreground objects can be assigned to distinct *COR*s. Right: Both *TO*s would be assigned to the same *COR* due to occlusion. 16

4.1. Visualization of the geometric transformations occuring in the pose estimation problem. 20
4.2. Simplified model of the pinhole projection assuming equivalent focal length in both directions and no skew. A point $\boldsymbol{x}$ in the camera frame is projected onto $\boldsymbol{x}'$ in the image plane. 21
4.3. Illustration of the reduced observation space. A point in the camera frame ${}^{c}\boldsymbol{p}_i$ is projected onto the unit sphere. The tangential plane corresponding to projected observation $\boldsymbol{v}_i$ is spanned by its nullspace vectors $\boldsymbol{r}_i$ and $\boldsymbol{s}_i$. 24

5.1. Some examples of augmented training data. Scenes: Outdoor 01 (top left), Outdoor 02 (bottom left), Outdoor 03 (top center), Outdoor 04 (bottom center), Outdoor 05 (top right) and Lab 01 (bottom right). 33
5.2. Loss and Validation Curves (mAP) during training of a RetinaNet Detector for 50 epochs. 34
5.3. Frame of the sequence **Lab 03** where ARDEA's track is lost. Real position indicated by dashed orange box. 36
5.4. **Lab 02**: Trajectories of ARDEA in the image plane obtained from the two tracking approaches. Positions for matching timestamps are indicated by dotted lines. 38
5.5. Drawbacks of both tracking approaches: (a) Conventional Approach: Large number of detected objects (indicated by green bounding boxes) in scenes with changing illumination. (b) RetinaNet Detector: FP Prediction (indicated by red bounding box). 39
5.6. Farthest Detection (a) and exemplary FP detection (b) by RetinaNet Detector on sequence **Outdoor 07** 40

5.7. Example scenario for an artificially created trajectory and corresponding 2D observations including noises ($n = 10$). 42
5.8. Comparison of mean pose errors for varying number of trajectory points n. . . 43
5.9. Comparison of mean pose errors for varying noise levels k. 44
5.10. Comparison of standard deviations of the pose errors for varying noise level. . . 44
5.11. Measured position of ARDEA's camera frame in the frame of the LRU color camera split into x- y- and z-component (**Lab 02**). 45
5.12. Raw Odometry Readings are reduced to a lower number of readings yielding a reduced trajectory. 46
5.13. Translational component $\hat{t}_x$ of estimated pose and corresponding estimated standard deviation $\sigma_{\hat{t},x}$ for pose estimation on **Lab02**. 47
5.14. Translational component $\hat{t}_y$ of estimated pose and corresponding estimated standard deviation $\sigma_{\hat{t},y}$ for pose estimation on **Lab02**. 48
5.15. Translational component $\hat{t}_z$ of estimated pose and corresponding estimated standard deviation $\sigma_{\hat{t},z}$ for pose estimation on **Lab02**. 48

6.1. Prediction Output of RetinaNet Detector for a distant view of ARDEA for the far range image from chapter 1. 51

Bibliography

[1] A. Wedler, M. Wilde, A. Dömel, M. G. Müller, J. Reill, M. Schuster, W. Stürzl, R. Triebel, H. Gmeiner, B. Vodermayer, *et al.*, "From single autonomous robots toward cooperative robotic interactions for future planetary exploration missions", in *Proceedings of the International Astronautical Congress, IAC*, International Astronautical Federation (IAF), 2018.

[2] Y. Gao and S. Chien, "Review on space robotics: Toward top-level science through space exploration", *Science Robotics*, vol. 2, no. 7, 2017.

[3] *Arches projekt details*, `https://www.arches-projekt.de/projekt-arches/arches-projekt-details/`, (Accessed on 12/04/2020).

[4] M. J. Schuster, M. G. Müller, S. G. Brunner, H. Lehner, P. Lehner, A. Dömel, M. Vayugundla, F. Steidle, P. Lutz, R. Sakagami, *et al.*, "Towards heterogeneous robotic teams for collaborative scientific sampling in lunar and planetary environments", 2019.

[5] M. J. Schuster, M. G. Müller, S. G. Brunner, H. Lehner, P. Lehner, R. Sakagami, A. Dömel, L. Meyer, B. Vodermayer, R. Giubilato, *et al.*, "The arches space-analogue demonstration mission: Towards heterogeneous teams of autonomous robots for collaborative scientific sampling in planetary exploration", *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5315–5322, 2020.

[6] M. Miiller, F. Steidle, M. J. Schuster, P. Lutz, M. Maier, S. Stoneman, T. Tomic, and W. Stürzl, "Robust visual-inertial state estimation with multiple odometries and efficient mapping on an mav with ultra-wide fov stereo vision", in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 3701–3708.

[7] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf", in *2011 International conference on computer vision*, IEEE, 2011, pp. 2564–2571.

[8] S. Urban, J. Leitloff, and S. Hinz, "MLPnP - a real-time maximum likelihood solution to the Perspective-n-Point problem", *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 3, no. 3, 2016.

[9] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey", *ACM computing surveys (CSUR)*, vol. 38, no. 4, 13–es, 2006.

[10] M. Fiaz, A. Mahmood, S. Javed, and S. K. Jung, "Handcrafted and deep trackers: Recent visual object tracking approaches and trends", *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–44, 2019.

[11] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, "Deep learning in video multi-object tracking: A survey", *Neurocomputing*, vol. 381, pp. 61–88, 2020.

[12] C. G. Harris, M. Stephens, *et al.*, "A combined corner and edge detector.", in *Alvey Vision Conference*, Citeseer, vol. 15, 1988, pp. 10–5244.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[14] D. Comaniciu and P. Meer, "Mean shift analysis and applications", in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, IEEE, vol. 2, 1999, pp. 1197–1203.

[15] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 747–757, 2000.

[16] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters", in *2010 IEEE computer society conference on computer vision and pattern recognition*, IEEE, 2010, pp. 2544–2550.

[17] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters", *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 583–596, 2014.

[18] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection", in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, IEEE, vol. 1, 2005, pp. 886–893.

[19] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus, "Learning color names for real-world applications", *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, 2009.

[20] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking", *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 8, pp. 1561–1575, 2016.

[21] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking", in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4310–4318.

[22] F. Li, C. Tian, W. Zuo, L. Zhang, and M.-H. Yang, "Learning spatial-temporal regularized correlation filters for visual tracking", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4904–4913.

[23] A. Lukezic, T. Vojir, L. Čehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6309–6318.

[24] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection", *IEEE Access*, vol. 7, pp. 128 837–128 868, 2019.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[26] R. Girshick, "Fast R-CNN", in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[27] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks", in *Advances in neural information processing systems*, 2015, pp. 91–99.

[28] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector", in *European conference on computer vision*, Springer, 2016, pp. 21–37.

[29] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[30] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[31] ——, "Yolov3: An incremental improvement", *arXiv preprint arXiv:1804.02767*, 2018.

[32] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection", *arXiv preprint arXiv:2004.10934*, 2020.

[33] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection", in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[34] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[35] L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation", in *CVPR 2011*, IEEE, 2011, pp. 2969–2976.

[36] B. Triggs, "Camera pose and calibration from 4 or 5 known 3D points", in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, IEEE, vol. 1, 1999, pp. 278–284.

[37] C.-P. Lu, G. D. Hager, and E. Mjolsness, "Fast and globally convergent pose estimation from video images", *IEEE transactions on pattern analysis and machine intelligence*, vol. 22, no. 6, pp. 610–622, 2000.

[38] V. Garro, F. Crosilla, and A. Fusiello, "Solving the PnP problem with anisotropic orthogonal procrustes analysis", in *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, IEEE, 2012, pp. 262–269.

[39] F. Moreno-Noguer, V. Lepetit, and P. Fua, "Accurate non-iterative O(n) solution to the PnP problem", in *2007 IEEE 11th International Conference on Computer Vision*, IEEE, 2007, pp. 1–8.

[40] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem", *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[41] J. A. Hesch and S. I. Roumeliotis, "A direct least-squares (DLS) method for PnP", in *2011 International Conference on Computer Vision*, IEEE, 2011, pp. 383–390.

[42] Y. Zheng, S. Sugimoto, and M. Okutomi, "ASPnP: An accurate and scalable solution to the Perspective-n-Point problem", *IEICE TRANSACTIONS on Information and Systems*, vol. 96, no. 7, pp. 1525–1535, 2013.

[43] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi, "Revisiting the PnP problem: A fast, general and optimal solution", in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2344–2351.

[44] L. Kneip, H. Li, and Y. Seo, "UPnP: An optimal O(n) solution to the absolute pose problem with universal applicability", in *European Conference on Computer Vision*, Springer, 2014, pp. 127–142.

[45] L. Ferraz, X. Binefa, and F. Moreno-Noguer, "Very fast solution to the PnP problem with algebraic outlier rejection", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 501–508.

[46] L. Ferraz Colomina, X. Binefa, and F. Moreno-Noguer, "Leveraging feature uncertainty in the PnP problem", in *Proceedings of the BMVC 2014 British Machine Vision Conference*, 2014, pp. 1–13.

[47] W. Förstner, "Minimal representations for uncertainty and estimation in projective spaces", in *Asian Conference on Computer Vision*, Springer, 2010, pp. 619–632.

[48] N. Pitchandi and S. P. Subramanian, "Image uncertainty-based absolute camera pose estimation with fibonacci outlier elimination", *Journal of Intelligent & Robotic Systems*, vol. 96, no. 1, pp. 65–81, 2019.

[49] M. A. Mehralian and M. Soryani, "EKFPnP: Extended kalman filter for camera pose estimation in a sequence of images", *CoRR*, vol. abs/1906.10324, 2019. arXiv: 1906.10324.

[50] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction", in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, IEEE, vol. 2, 2004, pp. 28–31.

[51] Z. Chen, Z. Hong, and D. Tao, "An experimental survey on correlation filter-based tracking", *CoRR*, vol. abs/1509.05520, 2015. arXiv: 1509.05520.

[52] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[53] Y. Yang and G.-A. Bilodeau, "Multiple object tracking with kernelized correlation filters in urban mixed traffic", in *2017 14th Conference on Computer and Robot Vision (CRV)*, IEEE, 2017, pp. 209–216.

[54] P.-L. St-Charles and G.-A. Bilodeau, "Improving background subtraction using local binary similarity patterns", in *IEEE Winter Conference on Applications of Computer Vision*, IEEE, 2014, pp. 509–515.

[55] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.

[56] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[57] *Least-squares solution in presence of known covariance - Matlab documentation*, https://www.mathworks.com/help/matlab/ref/lscov.html, accessed on 11/06/2020.

[58] Å. Björck, *Numerical methods for least squares problems*. SIAM, 1996.

[59] D. W. Eggert, A. Lorusso, and R. B. Fisher, "Estimating 3D rigid body transformations: A comparison of four major algorithms", *Machine vision and applications*, vol. 9, no. 5-6, pp. 272–290, 1997.

[60] J. E. Mebius, "Derivation of the euler-rodrigues formula for three-dimensional rotations from the general formula for four-dimensional rotations", *arXiv preprint math/0701759*, 2007.

[61] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[62] *Least-squares (model fitting) algorithms - matlab & simulink - mathworks deutschland*, https://de.mathworks.com/help/optim/ug/least-squares-model-fitting-algorithms.html, (Accessed on 12/09/2020).

[63] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge", *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[64] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context", *CoRR*, vol. abs/1405.0312, 2014. arXiv: 1405.0312.

A
Derivations for Pose Estimation

A.1. Detailed Covariance Propagation for Static Camera

To obtain the equations for covariance propagation from odometry readings to uncertainties of the position in the camera frame, the error model (4.14) is plugged into the equations for the position of ARDEA in the camera frame (4.1). The linear error propagation in equation (4.17) are here derived in detail.
For ${}^c\boldsymbol{p}_i$, one obtains

$$ {}^c\hat{\boldsymbol{p}}_i - {}^c\delta\boldsymbol{p}_i = {}^c\hat{\boldsymbol{t}}_i - {}^c\delta\boldsymbol{t}_i \tag{A.1} $$

with ${}^c\hat{\boldsymbol{p}}_i = {}^c\hat{\boldsymbol{t}}_i \Rightarrow {}^c\delta\boldsymbol{p}_i = {}^c\delta\boldsymbol{t}_i$.

For ${}^c\boldsymbol{p}_{i-1}$, one obtains

$$ \begin{aligned} {}^c\hat{\boldsymbol{p}}_{i-1} - {}^c\delta\boldsymbol{p}_{i-1} &= {}^c\hat{\boldsymbol{t}}_i - {}^c\delta\boldsymbol{t}_i + {}^c\hat{\mathbf{R}}_i \left(\mathbf{I} + \lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times\right) \left({}^i\hat{\boldsymbol{t}}_{i-1} - {}^i\delta\boldsymbol{t}_{i-1}\right) \\ &= {}^c\hat{\boldsymbol{t}}_i - {}^c\delta\boldsymbol{t}_i + {}^c\hat{\mathbf{R}}_i {}^i\hat{\boldsymbol{t}}_{i-1} - {}^c\hat{\mathbf{R}}_i {}^i\delta\boldsymbol{t}_{i-1} + {}^c\hat{\mathbf{R}}_i \lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times {}^i\hat{\boldsymbol{t}}_{i-1} - {}^c\hat{\mathbf{R}}_i \lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times {}^i\delta\boldsymbol{t}_{i-1} \end{aligned} \tag{A.2} $$

With ${}^c\hat{\boldsymbol{p}}_{i-1} = {}^c\hat{\boldsymbol{t}}_i + {}^c\hat{\mathbf{R}}_i {}^i\hat{\boldsymbol{t}}_{i-1}$ and neglecting quadratic error terms it stays:

$$ \begin{aligned} -{}^c\delta\boldsymbol{p}_{i-1} &= -{}^c\delta\boldsymbol{t}_i - {}^c\hat{\mathbf{R}}_i {}^i\delta\boldsymbol{t}_{i-1} + {}^c\hat{\mathbf{R}}_i \lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times {}^i\hat{\boldsymbol{t}}_{i-1} \\ \iff {}^c\delta\boldsymbol{p}_{i-1} &= {}^c\delta\boldsymbol{t}_i + {}^c\hat{\mathbf{R}}_i {}^i\delta\boldsymbol{t}_{i-1} - {}^c\hat{\mathbf{R}}_i \lfloor {}^c\delta\boldsymbol{\phi}_i \rfloor_\times {}^i\hat{\boldsymbol{t}}_{i-1} \end{aligned} \tag{A.3} $$

With $\boldsymbol{a} \times \boldsymbol{b} = -\boldsymbol{b} \times \boldsymbol{a}$:

$$ {}^c\delta\boldsymbol{p}_{i-1} = {}^c\delta\boldsymbol{t}_i + {}^c\hat{\mathbf{R}}_i \lfloor {}^i\hat{\boldsymbol{t}}_{i-1} \rfloor_\times {}^c\delta\boldsymbol{\phi}_i + {}^c\hat{\mathbf{R}}_i {}^i\delta\boldsymbol{t}_{i-1} \tag{A.4} $$

This can be anlogously repeated for any ${}^c\boldsymbol{p}_{i-j}$.

A.2. Construction of Matrices for Covariance Propagation from VO Results

Construction of the Matrices $\mathbf{A}$ and $\mathbf{B}$ in equation (4.19) can be obtained by collecting terms in (4.17). For n odometry readings, they can be computed as follows:

- $\mathbf{C} \in \mathbb{R}^{3(n+1)\times 6}$ for covariance propagation in a filtering approach:

$$ \mathbf{A} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \mathbf{I} & {}^c\hat{\mathbf{R}}_i \lfloor {}^i\hat{\boldsymbol{t}}_{i-1} \rfloor_\times \\ \mathbf{I} & {}^c\hat{\mathbf{R}}_i \lfloor {}^i\hat{\boldsymbol{t}}_{i-1} + {}^i\hat{\mathbf{R}}_{i-1} {}^{i-1}\hat{\boldsymbol{t}}_{i-2} \rfloor_\times \\ \vdots & \vdots \\ \mathbf{I} & {}^c\hat{\mathbf{R}}_i \lfloor \sum_{k=1}^{n} \left(\prod_{l=1}^{k-1} \left({}^{i-l+1}\hat{\mathbf{R}}_{i-l} \right) {}^{i-k+1}\hat{\boldsymbol{t}}_{i-k} \right) \rfloor_\times \end{bmatrix} \tag{A.5} $$

- $\mathbf{B} \in \mathbb{R}^{3(n+1)\times 6n}$ for propagation of odometry uncertainties (Remark: matrices written outside of the brackets are multiplied to each row):

$$\mathbf{B} = {}^{c}\hat{\mathbf{R}}_i \begin{bmatrix} \boldsymbol{b}_1 & \boldsymbol{b}_2 & \boldsymbol{b}_3 & \boldsymbol{b}_4 & \dots & \boldsymbol{b}_n & \boldsymbol{b}_{n-1} \end{bmatrix} \tag{A.6}$$

with

$$\begin{aligned}
\boldsymbol{b}_1 &= \begin{bmatrix} \mathbf{O} \\ \mathbf{I} \\ \mathbf{I} \\ \mathbf{I} \\ \vdots \\ \mathbf{I} \end{bmatrix}, \boldsymbol{b}_2 = {}^{i}\hat{\mathbf{R}}_{i-1} \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \lfloor {}^{i-1}\hat{\boldsymbol{t}}_{i-2} \rfloor_\times \\ \lfloor {}^{i-1}\hat{\boldsymbol{t}}_{i-2} \rfloor_\times + {}^{i-1}\hat{\mathbf{R}}_{i-2} \lfloor {}^{i-2}\hat{\boldsymbol{t}}_{i-3} \rfloor_\times \\ \vdots \\ \sum_{k=2}^{n} \left(\prod_{l=1}^{k-2} \left({}^{i-l}\hat{\mathbf{R}}_{i-l-1} \right) \lfloor {}^{i-k+1}\hat{\boldsymbol{t}}_{i-k} \rfloor_\times \right) \end{bmatrix} \\
\boldsymbol{b}_3 &= {}^{i}\hat{\mathbf{R}}_{i-1} \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{I} \\ \mathbf{I} \\ \vdots \\ \mathbf{I} \end{bmatrix}, \boldsymbol{b}_4 = {}^{i}\hat{\mathbf{R}}_{i-1} {}^{i-1}\hat{\mathbf{R}}_{i-2} \begin{bmatrix} \mathbf{O} \\ \mathbf{O} \\ \mathbf{O} \\ \lfloor {}^{i-2}\hat{\boldsymbol{t}}_{i-3} \rfloor_\times \\ \lfloor {}^{i-2}\hat{\boldsymbol{t}}_{i-3} \rfloor_\times + {}^{i-2}\hat{\mathbf{R}}_{i-3} \lfloor {}^{i-3}\hat{\boldsymbol{t}}_{i-4} \rfloor_\times \\ \vdots \\ \sum_{k=3}^{n} \left(\prod_{l=1}^{k-3} \left({}^{i-l}\hat{\mathbf{R}}_{i-l-1} \right) \lfloor {}^{i-k+1}\hat{\boldsymbol{t}}_{i-k} \rfloor_\times \right) \end{bmatrix} \\
&\vdots \\
\boldsymbol{b}_{2n-1} &= \prod_{k=1}^{n} {}^{i-k+1}\hat{\mathbf{R}}_{i-k} \begin{bmatrix} \mathbf{O} \\ \vdots \\ \mathbf{0} \\ \mathbf{I} \end{bmatrix}, \boldsymbol{b}_{2n} = \begin{bmatrix} \mathbf{O} \\ \vdots \\ \mathbf{0} \end{bmatrix}
\end{aligned} \tag{A.7}$$

A.3. Computation of the Jacobian for Spherical Normalization

In chapter 4, it is stated that the Jacobian of the spherical normalization will have lower triangular form and is of the following form:

$$\mathbf{J}_{\boldsymbol{v}} = \begin{bmatrix} \mathbf{J}_{\boldsymbol{v}_i} & & & & \mathbf{0} \\ \mathbf{J}_{\boldsymbol{v}_{i-1}} & \mathbf{J}_{\boldsymbol{v}_{i-1}} & & & \\ \mathbf{J}_{\boldsymbol{v}_{i-2}} & \mathbf{J}_{\boldsymbol{v}_{i-2}} & \mathbf{J}_{\boldsymbol{v}_{i-2}} & & \\ \vdots & \vdots & \vdots & \ddots & \\ \mathbf{J}_{\boldsymbol{v}_{i-n}} & \dots & \dots & \dots & \mathbf{J}_{\boldsymbol{v}_{i-n}} \end{bmatrix} \tag{A.8}$$

This result will be derived in detail here.
Considering the observation equation (4.9) and introducing ${}^{(3D)}\boldsymbol{x}_{i-j}$ as the 3D observation before normalization, one obtains:

$$\begin{aligned}
\lambda_i \boldsymbol{v}_i &= {}^{(3D)}\boldsymbol{x}_i = {}^{c}\boldsymbol{t}_i \\
\lambda_{i-1} \boldsymbol{v}_{i-1} &= {}^{(3D)}\boldsymbol{x}_{i-1} = {}^{c}\boldsymbol{t}_i + {}^{c}\mathbf{R}_i {}^{i}\hat{\boldsymbol{p}}_1 = {}^{(3D)}\boldsymbol{x}_i + {}^{c}\mathbf{R}_i {}^{i}\hat{\boldsymbol{p}}_1 \\
\lambda_{i-2} \boldsymbol{v}_{i-2} &= {}^{(3D)}\boldsymbol{x}_{i-2} = {}^{c}\boldsymbol{t}_i + {}^{c}\mathbf{R}_i {}^{i}\hat{\boldsymbol{p}}_2 = {}^{c}\boldsymbol{t}_i + {}^{c}\mathbf{R}_i \left({}^{i}\hat{\boldsymbol{p}}_1 + \Delta\hat{\boldsymbol{p}}_{1,2} \right) \\
&= {}^{(3D)}\boldsymbol{x}_{i-1} + {}^{c}\mathbf{R}_i \Delta\hat{\boldsymbol{p}}_{1,2} \\
&\vdots
\end{aligned} \tag{A.9}$$

From this, we can conclude that every ${}^{(3D)}\boldsymbol{x}_{i-j}$ only depends on "previous" ${}^{(3D)}\boldsymbol{x}_{i-j}$'s. Together with

$$\boldsymbol{v}_{i-j} = \frac{{}^{(3D)}\boldsymbol{x}_{i-j}}{\left|\left|{}^{(3D)}\boldsymbol{x}_{i-j}\right|\right|} \tag{A.10}$$

we can follow that the Jacobian $\mathbf{J}_{\boldsymbol{v}}$ will have **lower triangular form**.
Let us construct this Jacobian step by step.

- **j = 0**
  The main diagonal block of size 3×3 of the first "block row" (corresponding to $\boldsymbol{v}_i$) of the Jacobian is

$$\mathbf{J}_{00} = \frac{\partial \boldsymbol{v}_i}{\partial {}^{(3D)}\boldsymbol{x}_i} = \frac{\partial}{\partial {}^{(3D)}\boldsymbol{x}_i}\left[\frac{{}^{(3D)}\boldsymbol{x}_i}{\left|\left|{}^{(3D)}\boldsymbol{x}_i\right|\right|}\right] = \frac{1}{\left|\left|{}^{(3D)}\boldsymbol{x}_i\right|\right|}\left(\mathbf{I}_3 - \boldsymbol{v}_i {\boldsymbol{v}_i}^T\right) \tag{A.11}$$

- **j = 1**
  The main diagonal block of size 3×3 of the second "block row" (corresponding to $\boldsymbol{v}_{i-1}$) of the Jacobian can be obtained analogously as

$$\mathbf{J}_{11} = \frac{\partial \boldsymbol{v}_{i-1}}{\partial {}^{(3D)}\boldsymbol{x}_{i-1}} = \frac{1}{\left|\left|{}^{(3D)}\boldsymbol{x}_{i-1}\right|\right|}\left(\mathbf{I}_3 - \boldsymbol{v}_{i-1} {\boldsymbol{v}_{i-1}}^T\right) \tag{A.12}$$

  The off-diagonal block is obtained (using the chain rule) as

$$\begin{aligned} \mathbf{J}_{10} = \frac{\partial \boldsymbol{v}_{i-1}}{\partial {}^{(3D)}\boldsymbol{x}_i} &= \frac{\partial \boldsymbol{v}_{i-1}}{\partial {}^{(3D)}\boldsymbol{x}_{i-1}} \cdot \frac{\partial {}^{(3D)}\boldsymbol{x}_{i-1}}{\partial {}^{(3D)}\boldsymbol{x}_i} \\ &= \frac{\partial \boldsymbol{v}_{i-1}}{\partial {}^{(3D)}\boldsymbol{x}_{i-1}} \cdot \mathbf{I}_3 \\ &= \mathbf{J}_{11} \end{aligned} \tag{A.13}$$

- **arbitrary j**
  For arbitrary j we can construct the corresponding diagonal block in the Jacobian as

$$\mathbf{J}_{jj} = \frac{1}{\left|\left|{}^{(3D)}\boldsymbol{x}_{i-j}\right|\right|}\left(\mathbf{I}_3 - \boldsymbol{v}_{i-j} {\boldsymbol{v}_{i-j}}^T\right) \tag{A.14}$$

  The off-diagonal (on lower half) blocks can be calculated by using the chain rule, which we have seen only multiplies identities. Thus, we have

$$\mathbf{J}_{j,(0\ldots j-1)} = \mathbf{J}_{jj} \tag{A.15}$$

- **Overall Jacobian** It follows that the overall Jacobian for the spherical normalization becomes

$$\mathbf{J}_{\boldsymbol{v}} = \begin{bmatrix} \mathbf{J}_{00} & \mathbf{0} & \ldots & \ldots & \mathbf{0} \\ \mathbf{J}_{10} & \mathbf{J}_{11} & \ldots & \ldots & \mathbf{0} \\ \mathbf{J}_{20} & \mathbf{J}_{21} & \mathbf{J}_{22} & \ldots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{J}_{n0} & \ldots & \ldots & \ldots & \mathbf{J}_{nn} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{\boldsymbol{v}_i} & & & & \mathbf{0} \\ \mathbf{J}_{\boldsymbol{v}_{i-1}} & \mathbf{J}_{\boldsymbol{v}_{i-1}} & & & \\ \mathbf{J}_{\boldsymbol{v}_{i-2}} & \mathbf{J}_{\boldsymbol{v}_{i-2}} & \mathbf{J}_{\boldsymbol{v}_{i-2}} & & \\ \vdots & \vdots & \vdots & \ddots & \\ \mathbf{J}_{\boldsymbol{v}_{i-n}} & \ldots & \ldots & \ldots & \mathbf{J}_{\boldsymbol{v}_{i-n}} \end{bmatrix} \tag{A.16}$$