Department of Physics and Astronomy

University of Heidelberg

Master thesis in Physics

submitted by

Karsten Krispin

born in Duisburg

2016

Bootstrapping of

Batch Sequence-to-Pointcloud Registration

for

Groundtruth Generation

This Master thesis has been carried out by

Karsten Krispin

at the

Interdisciplinary Center for Scientific Computing (IWR)

under the supervision of

Prof. Bernd Jähne

Abstract

Generating large amounts of reference data with very little manual intervention is required for performance analysis and recent learning methods. We present a method for bootstrapping the alignment of video sequences to laser scan data based on few manually annotated sequences with similar camera trajectories. The method consists of three steps: inference on a directed path graph constructed from image wise cross correlation is used for coarse alignment of two video sequences. Subsequently, we apply graph matching based on SIFT and spatial similarity to find corresponding control points between sequences and hence estimate the camera pose. The final step is a majority voting mechanism over all assignments, thereby regularizing over time. Thereby, the method is robust to visual clutter of dynamically moving parts as well as changes in illumination. Comparison of results with ground truth show that the performance of our method is en par with the manual annotation pipeline albeit significantly faster: our method was used to register a stereo and optical flow evaluation dataset in only 1/5th of the time that would be required for complete manual annotation.

Zusammenfassung

Für die Erstellung von umfangreichen Referenzdaten für *Performance Analysis* und Lern-Methoden ist es nötig, wenig zeitlich schlecht skalierende manuelle Interaktionsschritte zu benutzen. Wir zeigen daher eine Methode für die initiale Registrierung von Videosequenzen zu lasergescannten Punktwolken. Wenige manuell annotierte Referenzsequenzen werden zur Registrierung von Zielsequenzen mit ähnlicher Kamerafahrt und Lichtverhältnissen angewandt. Die entwickelte Methode besteht aus drei Teilen: Grobregistrierung ähnlicher Bildpaare mittels *Graph Inference* für einen Pfad durch eine Ähnlichkeitskarte. Anschließend werden zu jedem Bildpaar durch SIFT und geometrische Ähnlichkeiten mithilfe *Graphmatching* Zuweisungen von Kontrollpunkten aus ein Referenz- in ein Zielbild bestimmt. Dies registriert das Zielbild mit der Punktwolke. Abschliessend werden die Zuweisungen aus allen Bildern mittels Mehrheitsvotierungen zeitlich regularisiert. Die Methode ist dabei robust gegenüber dynamischen Verdeckungen und Änderungen in der Beleuchtung. Experimente mit *Groundtruth* zeigen, dass die Methode gleichauf mit händischer Annotierung ist, gleichwohl aber schneller: Die Registrierung eines solchen Datensatzes mit der Methode daurt bloß 1/5 tel der für händische Annotationen veranschlagte Zeit.

Contents

1 Introduction

Large scale datasets consisting of aligned intensity or color images and depth maps are becoming more and more important in computational vision. Recent insights gained in performance analysis research [9, 3, 14] indicate that algorithm results on smaller datasets may not be representative enough to generalize well to relevant applications such as autonomous driving. Additionally, (deep) learning [29, 17] requires large amounts of training data to outperform state of the art methods.

Besides the raw input data this also requires reference data that can be used to evaluate algorithms or to which learning algorithms can be applied. The reference information can be depth for stereo image pairs or classifications like pedestrians, bicycles or cars for images captured in automotive applications. Thus, a stereo algorithm can be compared to the reference depth and a learning algorithm can learn to classify the different objects in images given the reference labels.

The depth ground truth for stereo algorithms can be created as follows. Besides a stereo image pair one also needs information on the captured scene. This can be given by a precise geometric model of the captured objects. If the exact camera orientation with regard to these objects is known, the depth value for each pixel can be calculated as the distance of the camera to the respective object surfaces.

If the image of these objects was captured with a real camera, the true camera position is unknown and needs to be estimated. This can be done using control points. These are, for instance, corners in an image which have corresponding corners in the geometric model. The camera pose with regard to an object can then be estimated by minimizing the projection error. If the projection of these control points coincides with the image points, an estimate of the camera pose is found.

All these estimates are subject to errors and ambiguities [25, 9, 15]. To ensure sufficient accuracy of such real-world reference data, we always need manual interaction. However, manual interaction does not scale well in time which is in conflict with the need of large amounts of data.

One such large stereo and flow reference dataset for which the sequence registration needs to be performed has been acquired at the HCI. To overcome the problem of labor intensive manual control point annotation, this work proposes a semi-automatic bootstrap-technique that uses prior manual annotations of 3D-2D correspondences to automatically register further sequences with spatial constrained camera trajectories and similar illuminated recording conditions.

The proposed method is illustrated in figure 1.1 on the following page and consists of three parts: (a) video-based coarse alignment of the camera pose, (b) geometry-guided graph matching for finding correspondences in individual frames and (c) temporal regularization .

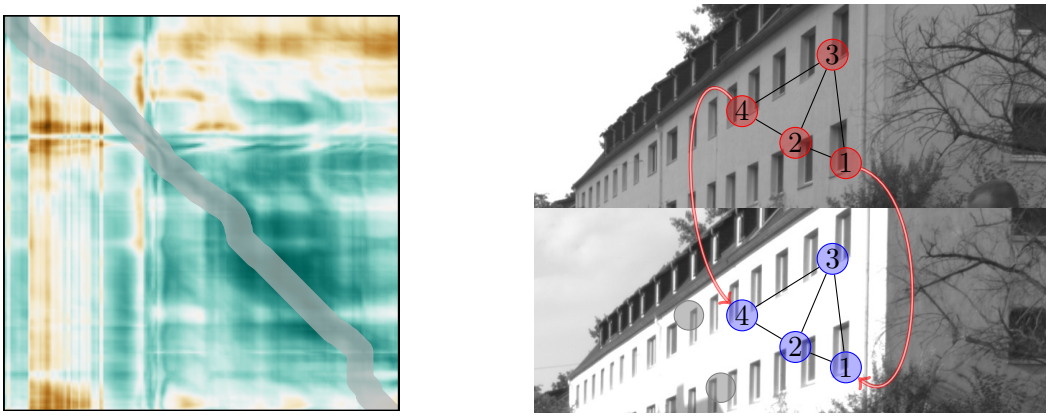


Figure 1.1: Overview of the developed method. A similarity map between all frames of a reference and target sequence is calculated, shown on the left. Frame pairs with highest similarity will be picked which evolves into the continuous gray path. After frame matching a feature matching will be applied (right) assigning control points from reference frame to target frame. Finally, the correspondences register the frames with the pointcloud.

In (a) we are using normalized cross-correlation (NCC) as similarity measure to temporally align frame pairs from a reference and target sequence. The frame pairs will be chosen by highest measure score. Due to repetitive elements in the sequence this score can be ambiguous. Therefore, a temporal regularization will be imposed by disallowing fast jumps between frame assignments pairs.

Next to frame matching follows (b) a feature matching step. Here, we want to find the annotated control points of a reference frame in the target frame again and use visual features to find correspondences. Due to highly repetitive structures in most images, a simple and unconstrained feature matching will almost always find wrong correspondences. Therefore, we construct a graph over all control points in a reference frame and make assignments in the target such that the geometric structure of the graph is most similar to the corresponding control points in the target frame.

The last step (c) is an outlier detection using unintrusive temporal regularization by means of majority votes.

These individual steps will be explained in detail in chapter 4. The general applicability of the proposed method will be evaluated in chapter 5. Prior to discussing the presented annotation method we give an overview on the properties of the processed dataset as well as related registration methods. This overview can be found in the next chapter 2 and is followed by a detailed derivation of the mathematical background in chapter 3.

2 Related Work

In this chapter, we focus on prior work and related methods that are important for this thesis.

The annotation method proposed in this thesis has a strong connection to a reference groundtruth dataset that has been acquired at the *Heidelberg Collaborative for Imaging* (HCI). For this reason, section 2.1 will give an overview of the used dataset in order to gain a better understanding for the requirements and difficulties of the dataset as well as the considerations made for the proposed method.

Next, we discuss related methods for video registration. In this context, we present the contributions of our method which can be found in section 2.2.

In addition, we examine requirements on the storage and processing framework in order to maintain data consistency with big datasets such as the one used in this work. These requirements are addressed in section 2.3.

2.1 Benchmark Suite

The developed method has been applied to a comprehensive stereo and optical flow reference dataset acquired prior to this work at HCI. The intention of the dataset is to provide a challenging benchmark suite for automotive applications. Data consist of a set of versatile sequences recorded in a stereo setup that show different features and difficulties.

The following will give an overview of the dataset and discuss a subset of its properties relevant for this thesis. As indicated above, our aim is to register different sequences from the dataset with one another. Hence, the following discussion also includes data-related implications on the registration method.

During data acquisition, emphasis has been put on variations of all kinds regarding the recording circumstances. These include different seasons, lighting conditions and dynamic objects like pedestrians. Examples for such different situations can be seen in figure 2.2 on page 15. The recording locations have been fixed to the L-shaped camera trajectories that can be seen in figure 2.1 on the following page. This was done to avoid combinatorical explosion of the different aspects of the dataset as would be the case for more liberally chosen trajectories. To our method this implies (I1) that the dataset contains strong positional and temporal constraints that can be exploited for correspondence matching.

The sequences have been captured during different seasons, different times of day and different weather conditions. In addition, traffic participants like pedestrians, bicyclists, car drivers and even pets have been employed to achieve dynamic scenes.

Figure 2.1: Bird's eye view of the scene in which sequences have been recorded. Most of them were acquired in the L-shaped path. The respective legs show a distance of approximately 230 m and 70 m for the longer and shorter, respectively.

These different aspects can be seen in figure 2.2 on the next page. All these features imply (I2) that many dynamic objects appear and disappear abruptly, giving rise to object occlusions. Furthermore, the radiometric variability has a strong influence on photometric correspondence matching as many objects may appear completly different due to strong exhibition to sun or sharp edges due to shadows as well as different levels of brightness.

Recording has been done with a resolution of 2560 px in width and 1080 px in height and a frame rate of 200 Hz. The high frame rate ensures that most motions within the image are subject to slow changes and the high spatial resolution ensures good track-ability of objects. However, the high spatial as well as temporal resolution have the downside of comprising redundant information. Processing the latter would not necessarily be beneficial, but time consuming. In order to achieve a fast annotation process it should be considered to (I3) skip as many as possible of these redundancies.

The geometric model according to which we calculate groundtruth consists of a large point cloud. It has been captured by a high resolution *light detection and ranging (LIDAR)* scanner, thereby leading to dense and precise pointclouds with a precision of 1 cm and a coverage of approximately 2500 points/m^2 on planar surfaces. However, such a high density of the pointcloud imposes problems related to the registration process. For instance, a corner may consist of several closely located points leading to ambiguous annotations. In combination with the former implications, this makes the registration task very challenging. In order to maintain accuracy it is inevitable to resort to human inspection for quality control. This implies (I4) that the annotation method should comply with human accessible features. Moreover, these registration

Figure 2.2: Overview of available sequences showing the different challenges. The sequences include different weather conditions like sun, rain and cloudy setups. In addition sequences were recorded during different times of the year. This results in strong photometric changes in the images like shadows or high exposures in sunlight. Further, we see dynamic objects like cars, pedestrians and dynamic occlusion of static geometry. Most of the target tracks that were annotated before are drawn from the window corners. They show periodic structures, are co-planar to each other and are often partially occluded.

features should be partially correctable by humans without starting the annotation process all over again.

In summary, we have addressed the different aspects of the dataset and identified implications for the development of a sequence registration method. In the following section, we proceed to giving an overview of available registration methods in prior publications that can be used to develop a registration method which suits the given dataset.

2.2 Related Registration Methods

The annotation method presented in this work is related to previous work on automatic image to point cloud alignment as well as video synchronization and alignment.

Corresponding work on *automated point cloud to image registration* exists in the context of aerial/remote sensing applications, for the alignment of multi view stereo and LIDAR.

One example are aerial image to cloud registration techniques [21, 13], an overview of which can be found in [22]. Methods that use aerial and ground viewing images have the advantage that in this context, imagery depth and texture edges coincide more frequently than in the general case such that a heuristic which correlates depth and intensity features works well [13].

In [31], the camera is still placed on plane, albeit observing the scene from an oblique angle. The approach here is to first compute depth maps from *moving camera stereo* and then using ICP to align the resulting images. However, Zhao et al. also require manual intervention to estimate the initial scale and pose with respect to the LIDAR point cloud. It should be noted that for all methods in aerial use cases, the observing camera is far removed from the scene, whereas in our case the camera is *in* the scene.

Another class of methods are concerned with aligning dense multi-view stereo geometry with LIDAR data [28, 19, 27]. The main approach is to first use ICP to obtain a coarse alignment of the two point clouds and then jointly solve for a multi view geometry and pose that is consistent with the LIDAR point cloud. These methods work well in their respective case because the cameras are observing a single object and the camera images have significant overlap, thus allowing for much more accurate geometry estimates than in our case of a forward-moving camera observing parts of the complete scene.

A different approach is *Video synchronization* for the temporal alignment of video sequences that have a similar camera trajectory [6, 5, 30]. The authors of [6] pose this as a database retrieval problem. However, their method is not suitable for our case since there are no established databases for our sequences. The authors of [5, 30] on the other hand make use of a graphical model for establishing corresponding frame assignments. Our coarse registration step conceptually borrows ideas from [5] while at the same time using different data terms and regularizers to allow for much

faster alignment without sacrificing accuracy.

A further field for image registration is the use of feature matching [20, 2, 18]. Feature matching denotes finding 2D correspondences from one image in another image. Features that can be matched usually include points in the image with large derivatives [26], such as corners. For the comparison of feature points there are two possibilities. The first way is patch-matching [18] where two points are compared to each other using visual patches. These patches, however, are not invariant against rotation or translation. To overcome this issue, a second way is to develop and compare features of the patches that are invariant against affine transformations [20, 2]. However, both methods are prone to mismatching for repetitively similar image patches.

To solve these ambiguities, the authors of [23] developed a feature matching algorithm based on graph matching. Here, possible matches are found by constructing a graph from a certain feature of a reference image and finding correspondences in target images such that the graph exhibits the same geometric relations. We use a similar concept in the feature matching step. However, the construction of the graph as well as the node and edge costs are different from [23], allowing for a more intuitive parametrization and to leave out single assignments if no possible matches can be found.

Image registration is also required in film industry. Here, one application is matching video sequences with synthetic special effects. Synchronization methods employed to this end need to be accurate or else the results will look unrealistic. Leading commercial tools such as NUKE and PFTrack offer sophisticated user interfaces for manual annotations to improve estimates of camera to point cloud pose. This can be taken as an indicator that manual interaction is required for accurate results in the registration process as is the case in our method as well.

In [15], the authors worked on the same dataset as presented in section 2.1 on page 13. In their work, they developed a method to annotate 2D-to-3D correspondences by sole use of manual work. The drawback of this approach lies in the fact that manual annotation is very time consuming.

This bottleneck will be addressed by this thesis. The annotation method presented in this work improves the pipeline of [15] by including manually bootstrapped control-points into a semi-automatic annotation process.

2.3 Data Storage and Processing Framework

In the former discussion we provided an overview of the dataset used in this work as well as related work concerning the registration step. Besides the dataset itself, a project structure definition and programming API have been developed to deal with the required huge amounts of data. The following gives a motivation for the usage of this framework within this work.

The dataset presented in section 2.1 contains approximately 200 sequences, each comprising 10000 stereo frame pairs per sequence. The required storage space

consists of approximately 15 TB which is mostly taken up by the images, followed by calibration data and point clouds. One sequence contains approximately 60 GB merely for the stereo image pairs. Loading a complete sequence into memory is not possible on any computer used for the present work. Therefore, only parts of the sequences can be loaded into and computed in memory. This requires caching implementations for every operation.

Besides the stereo pairs, a bundle of different meta information are stored, for instance the camera intrinsics, uncertainties and time of recording. Each frame comprises several thousand feature tracks of which only a small portion is used as control points in later course. The control points will be selected with the help of several tools and can be linked to different point clouds. This is required to happen iteratively as we usually need manual interaction. Taking care of all this meta information and the storage is a non-trivial task. Particularly when several tools written by different people interact, data consistency is a big challenge for a reference data set.

To tackle these storage problems in a consistent way, a framework called *H_O0KERS* [16] has been developed besides the mentioned dataset. The framework consists of two parts:

First, it comprises a well-documented project structure based on HDF5 [10]. In addition, a c++ programming interface to access the project files has been implemented. The interface exposes all objects as frontend/backend structures. This separation ensures that only one backend instance exists at a time and that we obtain proper caching for all objects. The first feature makes sure that access is possible in a consistent way, even in asynchronous applications like GUI operations. The second feature ensures that storage intensive objects like images can be passed around by reference without sacrificing performance of the application. Loading data is performed on demand and generated data instances disappear as soon as they are not needed anymore without requiring the framework user to implement caching over and over again. Caching also ensures that persistent storage is detached from frontend logic and avoids project file corruptions in case of an application crash. Concluding, the presented framework offers means for rapid prototyping combined with data consistency.

Second, the framework comprises the implementation of the pipeline steps that are used to generate groundtruth for the presented dataset. The different pipeline steps all access the data via the discussed interface. This ensures fast access to data as well as data consistency. Our annotation implementation therefor also makes heavy use of the formerly discussed framework and has been injected into *H_O0KERS* as an additional tool.

3 Background

This chapter introduces important mathematical concepts that will be referred to throughout this thesis. We start with the *pinhole camera model* in 3.1. This model is implicitly used in chapter 4 to model 3D-to-2D correspondences. The subsequent section 3.2 focuses on deriving camera poses using the pinhole camera model whereas section 3.3 covers methods for estimating the uncertainty of derived camera poses and introduces a means for comparing the results in a statistically sound manner.

3.1 Pinhole Camera Model

In the following, we make use of geometrical relations between 3D points belonging to point clouds and 2D feature points in images. To this end, we first need to understand the process of imaging, that is, mapping a 3D world onto a 2D image plane. This section elaborates on the mathematical model that is used to describe such a process.

A camera usually consists of an image sensor and a lens. The lens can be identified as a transformation that maps world points onto an image sensor. The image sensor in turn corresponds to the image plane. It maps light intensities of world points onto a logical grid, namely the pixels of an image.

The general case of mapping world points $X \in \mathbb{R}^3$ to image points $Y \in \Omega \subseteq \mathbb{R}2$ is described by the so called *point spread function*

$$\mathrm{PSF} : \mathbb{R}^3 \times \Omega \to \mathbb{R} \tag{3.1}$$

with

$$\int_\Omega dY \, \mathrm{PSF}(Y) = 1 \tag{3.2}$$

describing how the light intensity of one point X is distributed onto the image plane Ω. The PSF fully characterizes the imaging system. It can take the photo sensitivity of the image sensor into account, including wavelength, geometric and chromatic aberrations of the lens as well as diffraction of the aperture.

In this work, we are only interested in the projective position of a point X. Hence, we do not need to consider the complete physical process of imaging. This allows the PSF to be reduced to a *delta distribution* on the image plane Ω for every X, which has the following signature:

$$\pi : \mathbb{R}^3 \to \mathbb{R}^2 \tag{3.3}$$

By choosing such a function we neglect the wave nature of light and its diffraction effects and require the observed object to be in focus. Neglecting diffraction is

possible when the observed object is large compared to the wavelength. The object is sufficiently focused when the object distance is large compared to the focal length of the lens. Both holds for usual vision applications.

Camera lenses still produce distortion of image points, which leads to displacements of the projected points Y in the image plane. These distortions occur due to imperfect lens geometries or chromatic aberrations and are usually found in the form of radial distortions.

Such distortions can be corrected by modeling the projective properties of the lens, usually with the help of radial and tangential polynomials [12]. The polynomial coefficients are found using a calibration target with known geometry and comparing the distorted projection of the object with the true geometry. Further information about parameter estimation can be found in section 3.2.

All sequences presented in section 2.1 on page 13 have undergone such a calibration process. The images were rectified beforehand such that straight lines in the scene appear as straight lines in the images again. This allows us to neglect the distortion effects and leads to the *Pinhole Camera Model*:

$$\begin{pmatrix} Y_x \\ Y_y \\ 1 \end{pmatrix} = \pi_\theta(X) = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} H(\theta) \begin{pmatrix} X \\ 1 \end{pmatrix} \tag{3.4}$$

Here, $H : \mathbb{R}^4 \to \mathbb{R}^3$ is an operator, consisting of a homogeneous transformation followed by perspective divide. It maps the world coordinates X into the camera frame. Within camera frame, the image plane coincides with the xy-plane. The focal lengths (f_x, f_y) and the principal point (c_x, c_y) eventually transform the natural position into logical pixel positions. The focal length and principal point are called the *intrinsic parameters* and are obtained from camera calibration. The still arbitrary vector θ is called *camera extrinsics* and parametrizes the homogeneous transformation. The form of parametrization will be discussed in 3.2.

Given the pinhole camera model, it is possible to compute the projective position Y in the camera plane Ω of a world point X. The model is determined up to the camera extrinsics θ which parametrize the position and viewing direction of the camera within the world frame.

3.2 Parameter Estimation

In the former section, we have explained how to find the projective position of a 3D point onto an image sensor. Given the camera extrinsics and intrinsics, we can now compute the *forward problem* for every point X yielding the projection Y.

Many real-world processes are modeled as such a forward problem

$$F : \mathbb{R}^M \to \mathbb{R}^N, x \mapsto y = F(x) \tag{3.5}$$

However, this is not always what is required. Often, an observation $\hat{y}$ is given and the task is to find the parameter $\hat{x}$ which led to this observation. This yields the so

called *backward problem*:

$$F^{-1} : \mathbb{R}^N \to \mathbb{R}^M, y \mapsto x = F^{-1}(y) \tag{3.6}$$

Essentially, we hence need to find the inverse of our model F. In the following discussion, we illustrate how to find such inverse mappings.

In the above sections, we reasoned that our main task is to register sequences to a pointcloud. In the present case, we want to find camera poses θ using correspondences of 2D image features and 3D landmarks. To this end, we are given some set of mappings $(X_1, y_1), \ldots, (X_K, y_K)$ of 3D world points X_i and their respective projection Y_i according to equation (3.4) on the preceding page. We now effectively need to solve the backward problem to obtain θ.

When solving the inverse problem, we encounter a number of difficulties. First, finding a closed form of the inverse mapping is only straight forward for bijective linear problems, but often not possible at all. In the model defined by equation (3.4) on the facing page, the homogeneous matrix depends on the parameter θ. Hence, this model exhibits a non-linear behaviour and furthermore is not bijective. Thus we cannot find θ using only one correspondence. In order to assemble enough information for obtaining a well-defined solution, the correspondences need to be linearly independent and as such they should not be co-planar. Secondly, all measurements of X and Y are subject to errors.

Theoretically, it would be possible to solve the inverse problem with only four corresponding pairs [8]. However, this requires the points to be non-co-planar. In addition, measurement errors can have a huge impact on the solution. Hence we can expect the results calculated with four corresponding pairs to have a poor quality. When solving for the camera pose, we need many more correspondences then required for the analytical solution in order to get rid of the measurement errors and obtain a robust solution.

To this end, we formulate the backward problem as an optimization problem, effectively making use of the well-defined forward model. This leads to a *Nonlinear Least Squares Problem* of the following form:

$$\hat{x} = \underset{x}{\operatorname{argmin}}\ \phi(x,\ y_1, \ldots) = \underset{x}{\operatorname{argmin}}\ \sum_i \left(y_i - F_i(x)\right)^2 \tag{3.7}$$

This so-called *objective function* $\phi : \mathbb{R}^K \to \mathbb{R}$ reduces the backward problem to a minimization problem. Solving for x

$$\nabla_x \phi(x,\ y_1, \ldots) = 0 \tag{3.8}$$

leads to the desired parameters.

Finding derivatives of the forward model usually is straight forward. Hence, the latter set of equations can easily be computed from the forward model in an analytical way. Computing gradients of the inverse problem is much more accessible for computational methods than finding analytical solutions. Hence, for a large set of correspondences it is convenient to use numerical methods that minimize the objective function and lead to an estimate $\hat{x}$.

Pose Parametrization In the case of the pinhole model equation (3.4) on page 20, we need to find the homogeneous transformation H that can be expressed by a $\mathbb{R}^{4\times4}$ matrix. In general, this transformation is a composition of one translation and one rotation. An arbitrary rotation matrix can be described by three parameters, a translation by a further three. Hence, not all of the coefficients in the transformation matrix are independent of each other. Solving for the twelve matrix entries would not be a sparse parametrization of the transformation.

Therefore, we choose our parametrization θ to be

$$\theta = (t_x, t_y, t_y, r_x, r_y, r_z) \tag{3.9}$$

with $\vec{t}$ being the translation vector and $\vec{r}/|\vec{r}|$ the rotation axis with the rotation angle $|\vec{r}|$. This rotational parametrization is called *angle-axis representation* and the corresponding vector $\vec{r}$ is called *rotation vector*.

The chosen representation not only removes the redundancies of the matrix elements, effectively constraining the matrix entries to unit vectors, but as a side effect also reduces the number of derivatives that need to be computed for equation (3.8) on the preceding page.

Regularization The fact that not only the solution, but also many approximate solutions will be found as local minima in the optimization problem defined in equation 3.7 makes it very hard to find the global minimum. This is by way of example shown in figure 3.1a on the next page. Here, $\hat{x}$ is the desired solution. However, the starting guess of x significantly influences whether $\hat{x}$ will be reached. The illustrated minima can also be local minima of a larger objective function. In this case, we need to avoid getting stuck in any of the local minima.

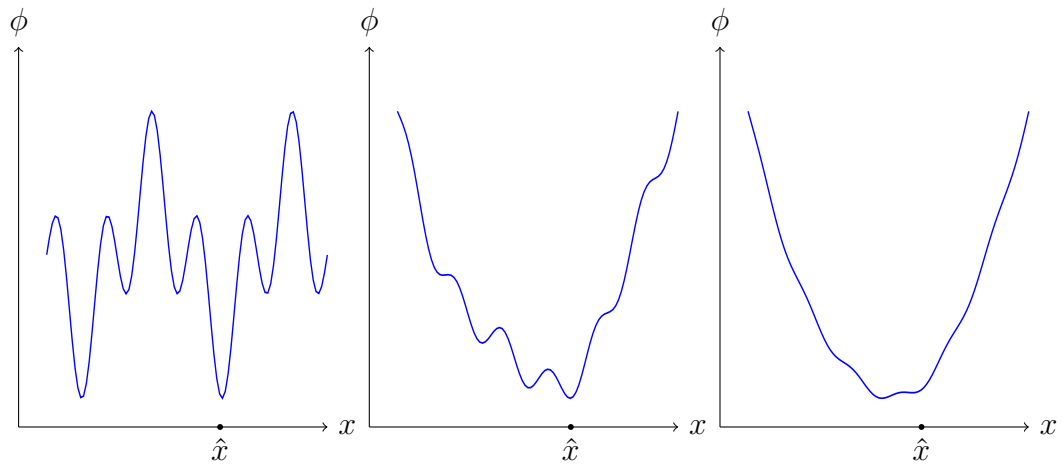
To this end, we add further constraints to the solution, thereby effectively reducing the solution space. Adding constraints is done by means of *regularization*.

In the following example, we want our solution to be as near as possible to $x = 0$. We therefore add a term cx^2 to ϕ and thus penalize solutions that are far away from the constraint. This is illustrated in figure 3.1b. We can see how a unique solution evolves and that the local minima are deformed to plateaus, avoiding the minimization task to get stuck too early.

However, if the objective function is over-regularized, the solution will tend towards the minimum of the regularization instead. This case is illustrated in figure 3.1c and demonstrates that the selection of the regularization weights is a crucial for the solution and always needs to be carefully tuned to the respective problem.

3.3 Uncertainty Estimation

The former discussion dealt with projection models that allow to calculate the projected position Y of a 3D landmark X given a camera pose θ. Next, we discussed how to find a camera pose given a set of 2D/3D correspondences and using the pinhole



(a) Objective Function (b) Regularized (c) Overregularized

Figure 3.1: Influence of regularization on the objective function. On the left is shown an example for a pure objective function, in the middle with added regularizer. On the right is shown a overregularized objective function. The pure objective has many equal global minima, combined with local extrema. Regularization adds further contraints to the solution and so favoring only one of the many possible minima. However, if the objective is overregularized, then the solution will be governed by the regularizer.

camera model. We reasoned that the corresponding pairs exhibit measurement errors regarding their positions which also affect the solution θ.

In the following, we aim to quantify the effect of the measurement uncertainties on the solution by using error propagation. To this end, we first show how the least squares problem relates to random variables and then discuss how to propagate observation errors in the region of the solution θ.

We start again with equation (3.7) on page 21. The sum over all squared differences can be rewritten using the scalar product

$$\sum_i (F_i(\theta) - y_i)^2 = (F(\theta) - y)^T \cdot (F(\theta) - y) \tag{3.10}$$

We can compare this to the definition of the probability density function of a multi-variate Gaussian distribution

$$f_X(x) \propto \exp\left(-\frac{1}{2}(\mu - x)^T \cdot \Sigma^{-1} \cdot (\mu - x)\right) \tag{3.11}$$

and see that we can identify the model values $F(\theta)$ as mean value μ and the observations y as one realization of a standard normal distribution. When minimizing the least squares errors, we effectively find a model $F(\theta)$ that is most likely to yield the observations y under the assumption of a normal distributed error process.

Having identified the least squares problem as a *maximum likelihood estimator* with normal distributed error processes, we ask for a means to compare different solutions to the problem by taking their uncertainties into account.

We can motivate such a distance which uses the gaussian density function (3.11) by focusing on the exponential argument. Apart from the factor 1/2 and the covariance matrix Σ, the term resembles the squared euclidean distance. Omitting the factor 1/2 leads to the *Mahalanobis Distance*

$$m = \sqrt{(\mu - x)^T \cdot \Sigma^{-1} \cdot (\mu - x)} \tag{3.12}$$

which normalizes the difference of a realization x to the mean value μ by the variance Σ.

Given a random process, two independent solutions θ_1 and θ_2 are likely to deviate from each other. We can measure this deviation by assuming that they originate from the same true underlying solution ν and construct a composition

$$x = \theta_1 - \theta_2 \tag{3.13}$$

that has a mean value of

$$E[x] = E[\theta_1] - E[\theta_2] = \nu - \nu = 0 \tag{3.14}$$

and variance of

$$\begin{aligned}
\Sigma &= E[(x - \overbrace{E[x]}^{0}) \cdot (x - \overbrace{E[x]}^{0})^T] \\
&= E[(\theta_1 - \theta_2) \cdot (\theta_1 - \theta_2)^T] \\
&= E[\theta_1\theta_1^T] - E[\theta_1\theta_2^T] - E[\theta_2\theta_1^T] + E[\theta_2\theta_2^T] \\
&= E[\theta_1\theta_1^T] - E[\theta_1]E[\theta_2^T] - E[\theta_2]E[\theta_1^T] + E[\theta_2\theta_2^T] \\
&= E[\theta_1\theta_1^T] - \nu\nu^T + E[\theta_2\theta_2^T] - \nu\nu^T \\
&= \Sigma_1 + \Sigma_2
\end{aligned} \tag{3.15}$$

where the Σ_i are the covariance matrices of the solutions θ_i. Plugging (3.13) and (3.15) into the Mahalanobis distance yields

$$c = \sqrt{(\theta_1 - \theta_2)^T \cdot (\Sigma_1 + \Sigma_2)^{-1} \cdot (\theta_1 - \theta_2)} \tag{3.16}$$

as a metric to compare two solutions θ_1 and θ_2 for statistical consistency.

In order to compare two solutions, we are hence left with the task of finding their covariance matrices Σ_i.

The backward problem is solved by the least-squares formalism given in equation (3.7) on page 21. In the region of a solution θ_i, the nonlinear least squares problem can be linearized. In this case, we find a linear closed-form solution for the parameter estimation using *normal equations*. By applying linear error propagation

and solving for Σ_i we can find [1] that the covariance matrix of this problem is given by

$$\Sigma_i = \left(J^T \cdot \Sigma_y^{-1} \cdot J \right)^{-1} \tag{3.17}$$

where J is the Jacobian matrix evaluated at the solution θ_i and Σ_y is the covariance matrix of the observations y. The latter corresponds to known measurement uncertainties.

Obtaining the covariance matrices Σ_i completes the discussion about uncertainty estimation. Solutions θ_i can now be compared with the Mahalanobis distance in a statistically sound manner. We use this distance later in chapter 5 for performance analysis.

4 Registration Method

In the following chapter, we return to the the development of the proposed annotation method and the underlying statistical models as have been introduced in chapter 1.

From the introduction of the dataset in section 2.1 on page 13, we learned about implications that need to be taken into account for the registration task. We will consider these implications and explain the resulting structure of the proposed method in section 4.1.

The remainder of the chapter then discusses the individual steps of the annotation process in detail.

4.1 Outline

From the implications of the dataset discussed in section 2.1 on page 13, we learned that we have to expect challenging sequences that need to be registered. This leads to the fact that we cannot expect our method to work smoothly on all the different situations. In order to guarantee a complete and precise registration, the workflow requires I4 the possibility of manual intervention whenever it is necessary. Human interaction should be intuitive and effective as possible. The authors of [15] suggest control points for camera registration. These control points are chosen such that the 2D features in the image are visually well-perceptive features like window corners or gables of the surrounding houses in the scene. Due to their [15] initial annotations on the given dataset, the authors already created a workflow for manual annotations. This suggests that the method developed in this work should be compatible with the aforementioned manual annotation as a fallback for error correction. We expect manual correction to be necessary on local parts of the sequence. Hence, we also require our method to work as local as possible in order to guarantee as few as possible side effects due to manual interaction. Furthermore, the initially annotated sequences can be used as reference for further annotations. In conclusion, this suggests that the method should consider the same visual features as in [15]. Beside the compatibility with manual corrections, the most striking advantage is the reuseability of the already annotated sequences, thereby reducing the amount of additional initial human intervention to a minimum.

The registration of further target sequences T to a pointcloud L works by using the same corresponding 3D landmarks $\{l_k\} \subseteq L$ that are associated with tracks $r_m \in R_i$ in frames $R_i \in R$ of a reference sequence R. We then register a target sequence T by linking the matched 2D tracks $\{t_n\}$ of a target frame T_j to the same landmarks as the corresponding 2D matches $\{r_m\}$ of a reference frame R_i. Effectively, this

reduces the video-to-pointcloud registration to a 2D feature matching problem on a frame-wise basis. The principles of this feature matching will be further discussed in section 4.2.

Feature matching requires similar frames with similar features. Before we can match these features we hence have to find reference and target frame pairs that can be used for feature matching. A method that exploits I1 the strong temporal and spatial constraints that all sequences have in common will be presented in section 4.3.

Finally, the locality of the feature matching sometimes suffers from spatial ambiguities leading to wrong annotations. To overcome these limitations, we present an effective method for outlier detection using temporal regularization. The outlier detection does not affect the local inference on the feature matching. This post processing step is presented in section 4.4 on page 38.

In summary, the proposed annotation method consists of three steps, namely *Frame Matching*, *Feature Matching* and *Outlier Detection* for the reasons given above. The following sections will discuss the individual steps in detail.

4.2 Feature Matching

In the outline of the proposed method, we reasoned that video registration should be performed using feature matching. To this end, the following discussion presents different methods for feature matching and reasons about their useability regarding the implications that we have found in section 2.1 on page 13.

For the moment, we assume that we already have found a reference and target frame pair for which we can match image feature points. These frame pairs could be selected manually or by a further preprocessing step as discussed in 4.3. We further state that this preselection found a frame pair that has a sufficient visual overlap with a respective number of common features. Additionally, we assume that for reference and target frames, a set of feature points already exists that have been previously defined by a feature tracker or manual interaction. Given these preconditions, we can discuss the different matching procedures considered in order to find features from the reference point set in the target point set.

Patch Matching is a method comparing an image patch of the reference frame to an image patch of the target frame [18]. The patches that need to be compared are defined by a neighbourhood around the feature points we aim to find matches for. Comparison is then performed using a metric like the euclidean distance or convolution filters. The best match is chosen as the nearest patch in terms of the chosen metric. For the euclidean distance, this would be the minimum value, for convolutional filters it would be a maximum value. A crucial parameter for patch matching is the patch size. The selection of this parameter is not trivial: The size of the feature tracks varies with camera distance to the respective point. The features can have extensions as small as a few pixels or as large as a few ten pixels.

Furthermore, differences in camera translation and rotation can lead to different positions and scalings as well as rotations in the target image. The metrics compare the patches in a pixelwise way. Hence, even small shifts or differences in scale can lead to big differences in the metric value. This shows that the patch size needs to be selected carefully. Choosing the patch size too large would induce problems that arise due to the scaling or slight translations. Choosing them too small would not capture enough of the structure, thereby reducing the sensitivity. Due to the implication of high resolution I3, we can expect larger feature points in any case. Hence we are forced to choose larger patch sizes which would make the proposed matching method not invariant against scaling and rotation. Preliminary experiments proved this behaviour. Hence, we have to choose a method that is more robust against changes in rotation and scale.

Feature Descriptors are abstract representations of image patches using Gaussian filters, histograms etc. The goal of such an abstract representation is to achieve invariance against rotation and scaling [20, 2]. This is done by constructing feature vectors from different operations on the patches and again comparing these by a metric. In contrast to template matching, this concept does not compare single pixel values but values that are a product of many pixels which is the reason for the affine invariance. The best match is then the feature which is closest according to the metric. However, preliminary experiments showed that most of the annotated reference features were not found by the described method. Instead, this technique more often then not found neighbours or similar points. Note that many of the previously annotated tracks by [15] were window corners as can be seen in figure 2.2 on page 15. These features all look very alike. Hence, due to image noise and lighting differences we can understand the resultant mismatching of similar feature points. However, if the feature points are visually similar, we can expect this for the abstract feature vector as well. This can be put to use by looking for the reference feature point in the k *nearest neighbours* of the target features with respect to the abstract feature vector. In most cases it was seen to be sufficient to choose $k = 3$. In this case the correct control point was almost every time within the KNN set.

Graph Matching So far, we presented *Patch Matching* and *Feature Descriptors* as two possibilities to match tracks from the reference frame to a target frame. We identified the need for invariance against affine transformations and showed that patch matching is not robust against these transformations which led to the introduction of feature descriptors. Both presented techniques still show weaknesses at highly repetitive patterns like window corners. This only allows matching up to a set of possible candidates. Besides, we did not take into account the implication I2 of dynamic sequence elements. In this context, we concluded that due to dynamic elements in the sequences we have to expect object occlusion. This implies that not all tracks are likely to be found in a target frame. Making use of *KNN* search to find the closest matches, we will find completely unrelated correspondences in this case.

Figure 4.1: Illustration of the graph matching problem. A graph is constructed using annotated tracks in the reference image (left). We want to find assignments of the tracks in the target (right) such that the graph edges are most similar in reference and target frame. The arrows illustrate the assignment. The blue surfaces around the target tracks denote possible candidate sets PT_n. The grayish circles correspond to unrelated target tracks.

Thus, matching for individual feature points shows ambiguities in the assignment. We have to solve these ambiguities by taking further relations of the tracks into account. In the following, we discuss the development of a method that is capable of dealing with these ambiguities as well as object occlusion.

To this end, we recapitulate our annotation goals. We want to register a target frame T_j given a reference frame R_i. Both frames have corresponding feature tracks $\{r_n\}$ and $\{t_m\}$. Some of the reference tracks r_i are associated to a landmark l_k. We want to find feature correspondences $\{(r_n, t_m)\}$ between reference and target frames. We have seen that the assignments have to be made taking inter-feature relations into account in order to avoid ambiguities. For each found correspondence (r_n, t_m), we then assign the landmark l_k associated with each r_n to the corresponding t_m. By this linkage we finally register the target frame T_j with the pointcloud L.

From the former discussion, we have shown that using *feature descriptors* leads to a prechoice of possible candidates $\{t_m\}$ for each reference track r_n and call this set of candidates PT_n. Since we have to make allowance for occlusions due to I2, we extend each of the sets PT_n by a null target that denotes an unassigned correspondence.

We are thus left solving a combinatorical assignment problem. We solve this assignment problem by matching similar graphs using inter-feature relations as proposed by [23]. The construction of the respective graphs will be discussed in the following.

A graph in one frame consists of nodes and edges. Each node n is associated to one unary information. Every edge (n, m) connects two nodes n and m and carries information that relates both nodes. In our case a node corresponds to one track

in a frame. An example for such a graph can be seen in figure 4.1 on the facing page on the left side. For each node r_n in the reference frame, we want to find a corresponding t_m in the target frame such that all correspondences (r_n, t_m) resemble a graph in the target frame that is similar to the graph in the reference frame.

This leaves us with the need for finding a means to compare the similarity of the graphs and optimize the correspondences. The construction of this problem will be done as follows. For each r_n, we define a discrete label $a_n \in PT_n$ that denotes the target track to which the given r_n is linked and call this the assignment of r_n. Note that each prechoice PT_n contains a null track that does not belong to the target frame T_j, such that we find $a_n \notin T_j$. We want to optimize the set $\{a_k\}$ using a discrete *maximum likelihood estimation* (cf sec. 3.3). Hence, we have to formulate the similarity measure using the notion of *energy terms*. Together with the labels a_n we can define such energy terms for the unary and binary information of the nodes. In the following, $\vec{r_n}$ and $\vec{t_m}$ denote the position of the track within their corresponding frames.

We start with the unary term. From the former discussion, we learned that we need to match images that show visual similarity. In this case, we can expect that corresponding tracks show up in similar regions within R_i and T_j. Hence, we declare tracks to be similar if they exhibit similar positions within their respective frames. We also need to take the unassigned case into account. This label receives a default distance C_u. Effectively, this forces the optimization to leave an r_n unassigned when no tracks t_m are closer than the distance C_u. These considerations are expressed in equation (4.1).

$$\rho_n = \begin{cases} |\vec{r_n} - \vec{a_n}| & \text{for } a_n \in T_j \\ C_u & \text{else} \end{cases} \tag{4.1}$$

Next, we focus on the inter-feature relation. In the discussion of the unary terms we used the fact that the tracks exhibit similar positions in both reference and target frames. It is then reasonable to assume that the tracks also retain their relative positions to each other [23]. We will exploit this fact to define the *pairwise* or *binary* terms. Beside the general case where both labels a_n and a_m refer to valid tracks, we have to take care of special cases. First of all, the assignments a_n should be unique. We do not want different a_n to refer to the same t_m. We ensure the latter by giving this combination a high energy value. Secondly, if one of the a_n denotes no assignment, we give this constellation a fixed distance C_p; for both a_n and a_m unassigned we set the values to be $2C_p$. The former case ensures that no assignment will be made if no features within a given distance can be found. The latter favors solutions with at least one assignment by setting a higher clipping threshold here. The general case is then simply the difference between the tracks. Given the former

considerations we present the energy term in equation (4.2).

$$\phi_{nm} = \begin{cases} \infty & \text{for } a_n = a_m \in T_j \\ 2C_p & \text{for } a_n = a_m \notin T_j \\ C_p & \text{for } a_n \notin T_j \vee a_m \notin T_j \\ |(\vec{r_n} - \vec{r_m}) - (\vec{a_n} - \vec{a_m})| & \text{else} \end{cases} \tag{4.2}$$

In contrast to [23], the presented energy term compares the positions using euclidean distances. Using polar coordinates and splitting up this relation into angle and radial distance would once more induce problems regarding the scaling. Hence, we chose a measure that only consists of pixel distances and as such is intuitive to scale.

So far, we have presented the unary and binary terms used to compare two feature graphs. The remaining step is the assembly of these unary and binary terms into an optimization problem. The reference frame graph $G = (V, E)$ with its nodes V and edges E is constructed as follows. The set of nodes V consists of all tracks r_n that are connected to a landmark l_k. Each node r_n is connected to its K_r spatially nearest neighbours. The binary terms (4.2) are symmetric. Hence, $(r_n, r_m) = (r_m, r_n)$ are identical edges and we only add one of them to E. Since we pose the optimization problem as normal-distributed *Maximum Likelihood Estimation*, we have to square the energy terms and scale them by their variances. The scaling is given by σ_u and σ_p which can be identified as the expected standard deviations for track positions and inter-track relations. Finally, optimizing equation (4.3) for the respective $\{a_k\}$ yields the desired feature matching

$$U(\{a_n\}) = \sum_{n \in V} \frac{1}{\sigma_u^2} \rho_n^2 + \sum_{(n,m) \in E} \frac{1}{\sigma_p^2} \phi_{nm}^2 \tag{4.3}$$

This concludes how to construct a feature matching process that takes inter-feature relations into account using the construction of graphs. We next examine the proposed graphs with respect to their practical properties.

For the construction of edges, we learned that the spatially K_r nearest neighbours are chosen. Let there be cliques of tracks in the frames. The latter can happen for instance if tracks lie on different house facades, as can be seen in figure 2.2 on page 15. If the number of tracks per cluster is higher than K_r, we can observe the graph G to decompose into independent subgraphs. This decomposition has an impact on the optimization process as these subgraphs can effectively be optimized independently. This property can be exploited when camera pose differences lead to different reprojections of the points. In this case, the whole cluster can be shifted but the inter-cluster relations still show similar behaviour. If there were connections in between such clusters, this would lead to higher errors in the pairwise terms and thereby harmfully influence the optimization process. On the other hand, choosing a too small K_r leads to overclustering. This overclustering in turn leads to less inter-feature relations taken into account with accordingly more unfavorable assignments.

Due to the independence of subgraphs, we can still encounter completely wrong annotations. These can occur if two reference tracks in different subgraphs link to the same target tracks or when the track coverage is very sparse. In these cases no suitable geometric information is available and outlier are likely. Hence, the final step of the feature matching is a plausibility check using RANSAC [7]. Here we estimate the camera pose of the target frame using the made annotations. Obvious outliers are detected with RANSAC by comparing the reprojection of the landmarks with actual track position. If the reprojection for an individual point does not agree with the position within a range of Δr, we reject this annotation. If after this plausibility check there are still various tracks $\{r_n\}$ that all link to the same target t_n, all of the correspondences are dropped.

Another aspect are the the scales σ_u and σ_p for the unary and pairwise terms respectively. From section 3.2 on page 20 we learned about regularization and the effect of unsuitable weights, resulting in overregularization. Choosing wrong scales here reveals the same problem. We also learned about the duality of least-squares and Maximum-Likelihood-Estimation. By knowing that these scales correspond to uncertainties, they initially can be chosen intuitively by taking the observed variance. Considering this we can drastically reduce the search space for suitable parameters.

In this section, we presented a robust method to register a target frame T_j with a pointcloud L. This registration process makes use of a reference frame R_i that shows visual similarity to the target frame and was already registered with L before. The registration is the performed by matching annotated reference tracks r_n to tracks t_m in the target frame and thereby linking the target tracks t_m to landmarks l_k. This finally registers the target frame with the pointcloud. The proposed method makes use of graph matching. We presented how to construct such graphs and find best matches with respect to the graph. Finally, we presented a plausibility check for the matched tracks.

4.3 Frame Matching

So far, we have discussed the feature matching for a frame pair (R_i, T_j). The following section will be concerned with how to find such a frame pair.

From the introduction of the dataset we learned in the context of I1 that the sequences are recorded following one camera path through the scene. For two sequences that start in the same point and end in the same point, we can be certain that one frame pair (R_i, T_j) within both sequences matches sufficiently. We will exploit this high temporal and spatial correlation of the sequence in order to find an initial guess to the feature matching and discuss possible ways of establishing correspondence.

We first assume that the driving speed in both sequences is constant. Since the sequences start and end in nearly the same points, we can assume that the best match would be given by (R_i, T_i) where the index i denotes the same frame number within in the sequences. However, these assumptions are not met in our case. First

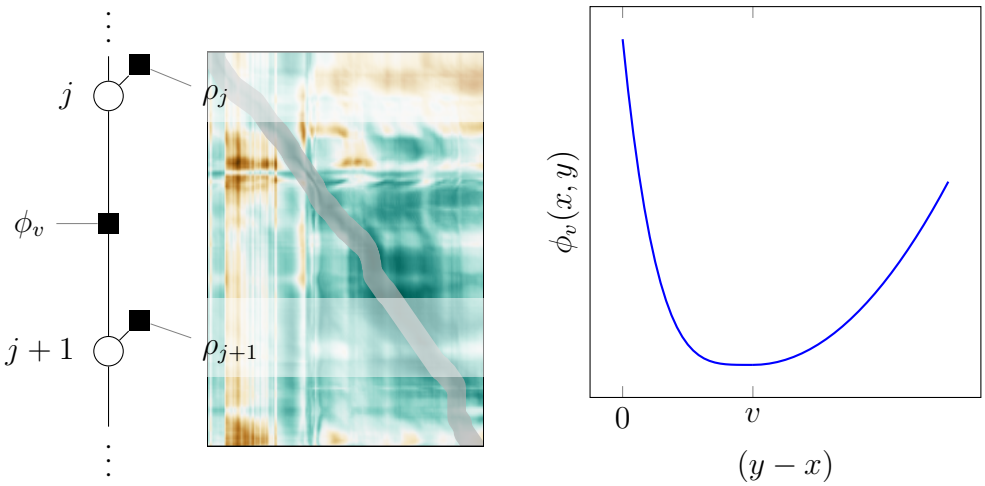


Figure 4.2: Visualization of the optimization problem. Left side shows the graph structure together with an example similarity map S. Each single-side potential ρ_j obtains it's values from one row of the S. The gray diagonal path corresponds to assignments that are optimal given this model. On the right one can see the qualitative form of the pairwise potential between two nodes j and $j+1$. Due to its' asymmetric form this potential discourages solutions backward in time.

of all, sequences do not necessarily start or end in the same points. Hence, the sequences have different lengths. Secondly, the vehicle sometimes slows down or even stops due to obstacles on the street. This implies that the chosen frame pair (R_i, T_j) will usually be out of sync such that $i \neq j$. For further reference, we call this method *diagonal* coarse alignment for reasons that we see later.

For the feature matching step, we require visually and spatially similar images and as such it is safe to assume that the chosen frames (R_i, T_j) show similar image content. In this case, we can compare the frame content using a simple pixelwise metric. Comparing each frame R_i to each frame T_j results in a similarity map S. The individual components are given by

$$S_{ij} = s(R_i, T_j) \tag{4.4}$$

where s denotes the similarity measure. We discuss a suitable choice for this measure later in this section. Given the measure s for each frame T_j, we can find a frame R_i by choosing the index i which shows the highest similarity according to s. We will later refer to this method as *unary max*. The described approach generally works well if the scene does not comprise more than one location with visual similarities. However, in the used dataset there are duplicate parts in the sequences that consequently exhibit visual similarity. Comparing the examples from figure 2.2 on page 15, we see similar images in between two of the houses. This means that similar values will arise

when driving past each house pair. The choice of j according to s is then dominated by image noise or different lighting effects and can lead to false assignments.

To solve these issues, we ultimately take the temporal coherence of both sequences into account as suggested by [5]. Frames are ordered within their sequence. Hence, given a certain correspondence (R_i, T_j) the neighbourhood can be approximated by $(R_{i\pm m}, T_{j\pm m})$. We also expect this temporal continuity within the similarity map S as illustrated in figure 4.2 on the preceding page on the left side. This suggests to fuse the similarity measure S with the temporal continuity to find the mappings (R_i, T_j), thereby exploiting spatial and temporal coherences between the respective sequences.

Path Inference In order to find a continuous path in the similarity map, we construct a Markov chain. In this case, we want to match target frames T_j against reference frames R_i. Hence, we identify each T_j with one node in the Markov chain. Each node has a label $a_j \in R$ that assigns one of the reference frames to the corresponding node T_j. Similar to the feature matching in section 4.2 on page 28, we pose the inference problem as an energy minimization problem. Hence we need to choose suitable unary and binary energy terms.

We start again with the unary terms. These terms consider information that is only related to one node. From the former discussion we have seen that we can find the best individual match using the similarity map S. This suggests to use S as unary term such that the resulting unary potential is given in equation (4.5).

$$\rho_j(i) = S_{ij} \tag{4.5}$$

The latter shows that the similarity map needs to be normalized such that the best unary match corresponds to the smallest value in ρ_j.

We now discuss the binary terms relating two frames T_j and T_{j+1}. Due to the continuity in time we can expect these frames to have solutions that are close to each other. We enforce this with a quadratic potential that penalizes solutions which encompass large gaps. Another property of the sequences is that for almost any sequence, the vehicle does not drive backwards such that we can suppress solutions backward in time. We achieve this by employing an asymmetric potential, thereby favoring solutions with $a_i < a_{i+1}$. Furthermore, we can expect the vehicles to have a known mean velocity larger than zero. This effectively pushes the solution forward in time and is expressed in the parameter v. An example for the pairwise term is given in figure 4.2 on the facing page whereas the definition of the pairwise term has the following form

$$\phi_v(x, y) = \begin{cases} (y - x - v)^2 & \text{for } (y - x - v) \geq 0 \\ (y - x - v)^4 & \text{for } (y - x - v) < 0 \end{cases}. \tag{4.6}$$

Having explained the construction of unary and binary terms, we finally have to assemble the optimization problem. The constructed factor graph is illustrated

in figure 4.2 on page 34 on the left hand side. The graph $G = (V, E)$ consists of nodes V for all target frames T_j and edges E that connect every (T_j, T_{j+1}) up to the T_{J-1}th frame. The final potential that is associated to the factor graph is given in equation (4.7). Optimizing U over the possible labels $\{a_j\}$ yields a path that is optimal given this model

$$U = \frac{1}{\sigma_u} \sum_{i \in V} \rho_i(a_i) + \frac{1}{\sigma_p^2} \sum_{(i,j) \in E} \phi_v(a_i, a_j) \tag{4.7}$$

From the definition of the graph we can now deduce properties that are relevant for productive use.

We have presented the parameter v as the mean velocity of the vehicle within the frames. The velocity must not necessarily be constant during the sequences. Obviously, the vehicle may perform local slowdowns or even stops. Hence, it can be necessary to determine the velocity in a local fashion. This can be achieved eg. using structure from motion [11], where the same principles of stereo triangulation are used to reconstruct scenes.

However, these estimates are subject to errors. Besides the local mean velocity we would hence also need to estimate the standard deviation σ_p in this case. If no local estimate is used, the mean velocity can still be set to a reasonable default value in the following way: In case that the similarity map S reveals no or far too little structure, we would like the path to follow a slope that corresponds to the diagonals of the similarity map. If the similarity map is a square matrix, the slope can simply be set to one. However, sequences are usually not equally long which needs to be taken into account. By setting v to the ratio between the number of reference and target frames we are able to assign v a slope that resembles the desired diagonal. This estimate to v can be used as a graceful fallback.

So far, we assumed that all frames from T are registered with a reference frame from R. Note implication I3 about high frame rates. There, we concluded that many of the frames will show high temporal redundancies. If we assume that the vehicle has a mean velocity of $8\,\frac{\mathrm{m}}{\mathrm{s}}$, corresponding to $30\,\frac{\mathrm{km}}{\mathrm{h}}$, we move by approximately 5 cm each frame. In the course of chapter 5, we will see that this corresponds to the resolution limit of the pose estimation. Even if we only process the annotation at 4 Hz, we will encounter a frame distance of 2 m. This shows the high redundancy of the dataset. Hence, it is highly adviseable to skip frames in order to keep computational cost low. Ideally, a subsampling should be chosen according to the speed of the vehicle. However, the former discussion shows that a constant subsampling should be sufficient when the number of subsamples is chosen with regard to the maximum velocity.

Similarity Measure and Pre-Filters In the former discussion we have presented how to construct an optimization problem taking frame similarities and temporal continuity into account. So far, we have delayed the choice of a suitable similarity measure. In the following, we address this point in terms of giving the reasons for the choice of the similarity measure. Besides, we discuss the for and against of prefilters.

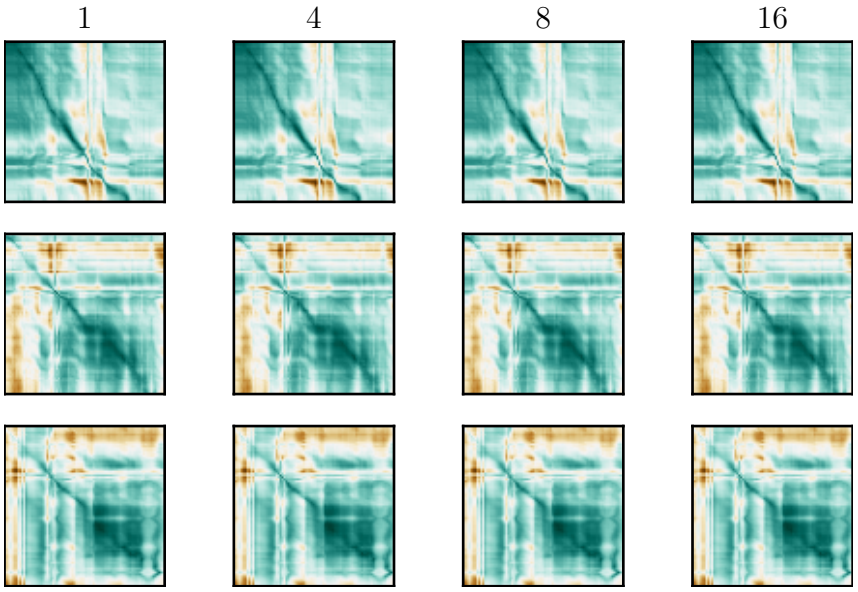


Figure 4.3: This figure contains several examples for prescaling before calculating a similarity map. From left to right the prescales are 1, 4, 8 and 16. From top to bottom are shown different sequence pairs.

In section 4.2 on page 28, we already reasoned about *Patch Matching* which is closely related to our current concern. In our case, patches are simply whole images without the need for search windows. The same objections that we have identified for patch matching then apply here as well.

The images that need to be compared show variations in lighting conditions, as addressed by I2. Hence, the same surface can be brighter or darker according to the current conditions. We would like to compare images independently of such variations. This implies that the measure needs to be a relative one. We aim to detect whether bright or dark regions match against each other. Effectively, this suggests to employ a pixelwise correlation of the images. In this course, we also need to normalize the images such that a correlation of 1 corresponds to a perfect match. The resulting measure is characterized by the following equation

$$S_{ij} = s(R_i, T_j) = \frac{\sum_{x,y} |R_i(x,y) \cdot T_j(x,y)|}{\sum_{x',y'} |R_i(x',y')| \cdot \sum_{x',y'} |T_j(x',y')|} \tag{4.8}$$

Applying this measure to all possible combinations of $\{R_i\}$ and $\{T_j\}$ then results in the aforementioned similarity map S.

Due to the high resolution of the frames it is questionable if the full image resolution is required to compute the similarity map. Therefore, we will discuss the effect of prescaling the images. We aim to understand how prescaling influences the contrast within the similarity maps. To this end, we show examples that were calculated using equation (4.8). Before applying the metric s we prescaled the images by fixed factors of $1, 4, 8, 16$. The results can be seen in figure 4.3. From this figure we deduce

that the prescaling has nearly no influence on the results. Scales of up to 16 show no qualitative difference. This confirms that the high spatial resolution carries much redundant information that we can omit for coarse matching by prescaling the input images.

4.4 Outlier Detection

So far, we have discussed the framewise feature matching in section 4.2 and the frame pair matching in section 4.3. Feature matching is applied framewise, thereby ignoring the temporal information of the sequences. The method was constructed this way to allow for local user interaction.

It is plausible to assume that the framewise matching will lead to not only correct correspondences. This is no problem for a supervised operation of the feature matching. However, when applying the pipeline to a whole sequence in an unsupervised way, it would be beneficial to still include a temporal regularizer. Here, we present an outlier detection method that can be used as a post-processing step to the aforementioned methods without interfering them.

The temporal connection between frames is mediated by the tracks t_m that usually appear individually in a set of consecutive frames $\{T_j, \ldots, T_{j+n}\}$. Hence, for each frame T_j we can remember the assignment of a track t_m to a landmark l_k. After the whole sequence T has been annotated we have a list of assignments $A_m = [l_u, \ldots, l_u, \ldots, l_v]$ for each track t_m. The final landmark assignment to each track t_m will be the landmark that occurs most often in A_m.

The presented outlier detection cannot be used to regularize the feature matching to a better global optimum. Still, it is capable of detecting and correcting local false assignments. Due to the continuity of tracks this outlier correction also helps to annotate frames where the framewise auto-annotator failed.

Together with the frame matching and the frame wise feature matching we presented a method to register a large set of similar video sequences to a pointcloud. The frame matching makes use of a temporal regularization over a similarity map. The framewise feature matcher uses geometric relations of tracks in a frame to find robust matches. The postprocessing step finally regularizes over time without interfering the former mentioned steps. Given these steps we are now able to register a large set of sequences using a small set of references.

5 Experiments

In this chapter we evaluate the applicability and performance of the proposed method. To this end, we examine the metric that is used for performance evaluation. This is followed by the introduction of groundtruth generation. The remainder of this chapter then evaluates the overall performance as well the single steps *Frame Matching*, *Feature Matching* and *Outlier Detection* and finally relate these insights to common difficulties of the scene in which the dataset has been recorded.

First we focus on the requirements on the performance measure with which we reason about the useability of the presented method. The presented semi-automatic registration method has been developed for the registration of video sequences to a pointcloud in order to generate grouthtruth like depth or optical flow. Manual registration is a labour-intensive task and as such is very time-consuming, such that it is adviseable to automate as much of the workflow as possible. Hence, we ultimately want to prove a speed-up in the registration process over manual annotation.

We start with the time consumption of manual annotation as proposed by [15]. Using manual annotation, it takes approximately two minutes per frame to register the frame using control points. Each of the sequences presented in 2.1 has approximately 10000 frames. Not all of the frames need to be annotated due to the feature tracks that persist in between several frames. Experience shows that it is sufficient to only consider every 20th to 50th frame, corresponding to 250 frames that need to be annotated per sequence. Hence, for one sequence, we need approximately 7 h of work – one working day. If one person were to annotate all the sequences, this person would be employed for one year. For the developed method this means that an error rate of 50%, which is very high on an absolute scale, would reduce the workload of one person by half a year. This would already be a huge advantage regarding time consumption.

The proposed method is semi-automatic. This implies that we expect human intervention in parts of the sequences. In order to prove significant time savings we need to count how many of the total frames are still required to be manually annotated. Hence, we need a performance measure that is capable of comparing the automatic annotation results to human annotation results and tell apart good from poor annotations. The way in which manual annotation (groundtruth) is created is going to be discussed later. In the following, we first want to focus on the mentioned annotation quality measure.

Performance Measure The central requirement for the proposed annotation method is that annotations resulting from manual interaction and annotations made using

this method exhibit the same accuracy in groundtruth generation. Groundtruth will be generated by reprojection (cf. 2.2), such that the key parameter is the camera pose that is used for reprojection. To this end, we compare the obtained camera pose of manual and automatic annotations of the very same frame with the following measure.

The pose p^i for any frame i is estimated by minimizing the re-projection error between annotated landmark points (3D) and the corresponding feature points (2D) as given in equation (3.4) on page 20 which is implemented as a least-squares problem.

Camera poses p^i are subject to uncertainties. These uncertainties can be a) systematic or b) measurement errors. Systematic errors arise due to mispicked landmarks or due to biases of the image feature tracker. Measurement errors on the other hand occur in the measurements using LIDAR or due to image noise leading to different track positions and can be considered to be Gaussian distributed [15]. Systematic errors occur in both groundtruth pose and automatic annotated pose because they are obtained using the same frames with same tracks and same landmarks. We compare the difference of both poses in which the same systematic errors will vanish anyway. This implies that we can expect processes dominated by Gaussian errors in the camera poses. It also suggests that neither the groundtruth pose nor the pose obtained by automatic annotation can be considered exact.

Furthermore, we can expect different uncertainties scales for different feature correspondences. Hence, for frames with a smaller number of correspondences or a more ambiguous constellation (co-planar points) we would expect a higher uncertainty also for the groundtruth pose. Naturally, the deviations between groundtruth and annotated pose will be higher in this case. This shows that we cannot expect the same scale of errors in all frames.

Thus, simply comparing two poses by euclidean distances in terms of position and rotational vector would induce a twofold problem: 1) Comparing rotations and translations independently of each other is not sufficient, because the values correlate 2) The different scales of uncertainties due to different feature correspondences would not be taken into account .

As suggested by [4] we address this by using the *Mahalanobis Distance* as given in equation (3.16) on page 24. This measure normalizes the deviations using a covariance matrix Σ, thereby taking correlation and different uncertainty scales into account. We call values obtained with the Mahalanobis distance *consistency values c*. These are used as our central quality measure. Here, small consistency values denote a good match and large values mark a poor match.

Since pose uncertainties can be considered normal distributed, values of more than three unit-distances ($3c$) denote inconsistent matches. In the latter case, we can be very certain that the pose deviations stem from wrong annotations and not from measurement uncertainties anymore and hence consider these annotations as false or *outliers*. Pose deviations by less than $3c$ can be considered insignificant and cannot be distinguished from deviations due to measurement errors. Therefore, we take these deviations to be consistent with the groundtruth pose and call them *inlier*.

Comparing consistency alone is not sufficient because consistency values are

normalized values and cannot be interpreted without the underlying covariance matrix Σ. In the following we refer to the diagonal entries as σ denoting the precisions, which is equivalent to the standard deviation in each component. The precisions are given by $\sigma = \sqrt{tr(\Sigma)}$.

Two arbitrary camera poses can still be consistent to each other with sufficiently large values in Σ. This is the case for instance if both camera poses deviate by 0.5m, but show precision values of 1km. Naturally, these poses are highly consistent within their uncertainties. Hence, consistency values are only meaningful if the underlying precisions are small enough with respect to the scene and exhibit similar values with respect to automatic annotation pose and groundtruth pose. During evaluation, we have to make sure that this requirement is met.

The pose estimation has been implemented as a least-squares problem and was modeled using the Ceres Solver [1]. It ships with a covariance analysis feature, directly exposing the covariance matrix Σ which corresponds to the optimized pose. For the observation uncertainties we set $\Sigma_y = 4 \cdot I$ corresponding to a standard deviation of 2 px. To filter obvious outliers, we solve equation (3.4) using a RANSAC scheme. Here, we set the inlier-region to 5px and are sampling twice times the number of available correspondences. More information about pose estimation and covariance analysis can be found in section 3.2 on page 20 and subsequent sections.

With the Mahalanobis distance, we have found a way of comparing two annotations in a statistically sound manner, yet one problem remains due to the ambiguity of the chosen pose parametrization. The rotation angle α of the pose (cf. equation (3.9) on page 22) can only be determined up to the periodicity of 2π. The same representation of the rotation matrix can also be attained by flipping the rotation axis and rotating by $(2\pi - \alpha)$. We can see that an arbitrary number of representations exists for the same rotation using the angle-axis representation. This should be illustrated by the two poses $(0, 0, 0, 0, 0, \pi)$ and $(0, 0, 0, 0, 0, 3\pi)$, both representing the same pose with a rotation around the z-axis by π. If we choose variances to be 0.01rad, these two poses would be highly inconsistent to each other although they represent the very same pose.

In order to compare two such poses using the *Mahalanobis Distance*, we have to normalize the rotational part first. The discussed ambiguity can be resolved by calculating the rotation matrix representing the present rotation. The matrix of such a unitary mapping is unique. Re-calculating the angle-axis representation from this unique matrix always leads to the same results, thereby normalizing the representation.

The covariances Σ (uncertainties of the camera pose) obtained from covariance analysis of the least-squares problem are not affected by this normalization because both normalized and unnormalized poses correspond to the same solution in terms of the objective function. Hence, the jacobian matrix is equal in equation (3.17) on page 25 and therefore the uncertainties are equal.

In summary, we have discussed the performance metric that is used to classify the annotation results. We chose the Mahalanobis Distance as a measure of *consistency c*.

Consistency in our case encodes how likely two obtained camera poses originate from a true underlying pose. Using this metric, we can compare automatic and manual annotations by comparing the obtained camera poses. This leaves us with the need for a means to generate groundtruth which will be discussed in the following.

Groundtruth In order to evaluate the overall performance of the proposed registration method it would be beneficial to have every sequence attached with groundtruth. However, this would require all sequences to be manually annotated and would render the proposed method useless. Hence, for quantitative evaluation, a representative subset of the sequences was chosen. The selected subset is illustrated in figure 5.1 on the next page.

The subset contains sequences for both driving trajectories, comprises different lighting conditions like strong sunlight with shadows (0_0059, 0_0068) and also different weather (0_0013). Besides, it includes sequences from two different seasons. This covers most of the challenge requirements imposed on the dataset.

Prior to any annotation, we assume a given set of tracks established between frames. In the following, we present the generation of such tracks.

In order to find interesting points in a frame, an eigenvalue cornerdetection method is used [26]. If an interesting point has been found, it will be tracked through consecutive frames using template matching. The patch scores are computed using pixelwise *squared absolute difference*, with a patch size of 21×21 and a search window size of 31×31. The parameter selection for the eigenvalue cornerdetection heavily depends on the sequence setup. Due to the different requirements of the dataset, a common choice for parameters in the feature detector is not possible. In order to guarantee high-quality tracks in all sequences, the parameters are fine-tuned. This is done as follows: From the introduction, we learned that the main task is to find window corners in target frames. Hence, the selection of the parameter is fine-tuned such that tracks are detected continuously around these window corners.

Following the tracking step, we establish a 2D-to-3D correspondences. This literally happens by hand-picking the correspondence according to visually well distinguishable features, such as window corners. The selection is done with a graphical user interface [15] rendering the tracks (2D) and landmarks (3D) clickable. Using these manually generated correspondences, the camera poses can be calculated, finally yielding the human-annotated groundtruth pose for each frame.

So far, we have reasoned about the requirements on the performance metric with which we evaluate our method and concluded that the notion of *consistency* compares results in a statistically sound manner. We then showed how groundtruth is created in order to compare the results using pose *consistency*.

The remainder of this chapter is organized as follows. An overview of the performance of the proposed annotation method is given in section 5.1 on page 44. There, we assess the general usability by applying the method to the set of test sequences introduced in section 5 and show that the requirements on the performance metric are met. Besides, we address general limitations of the method in section

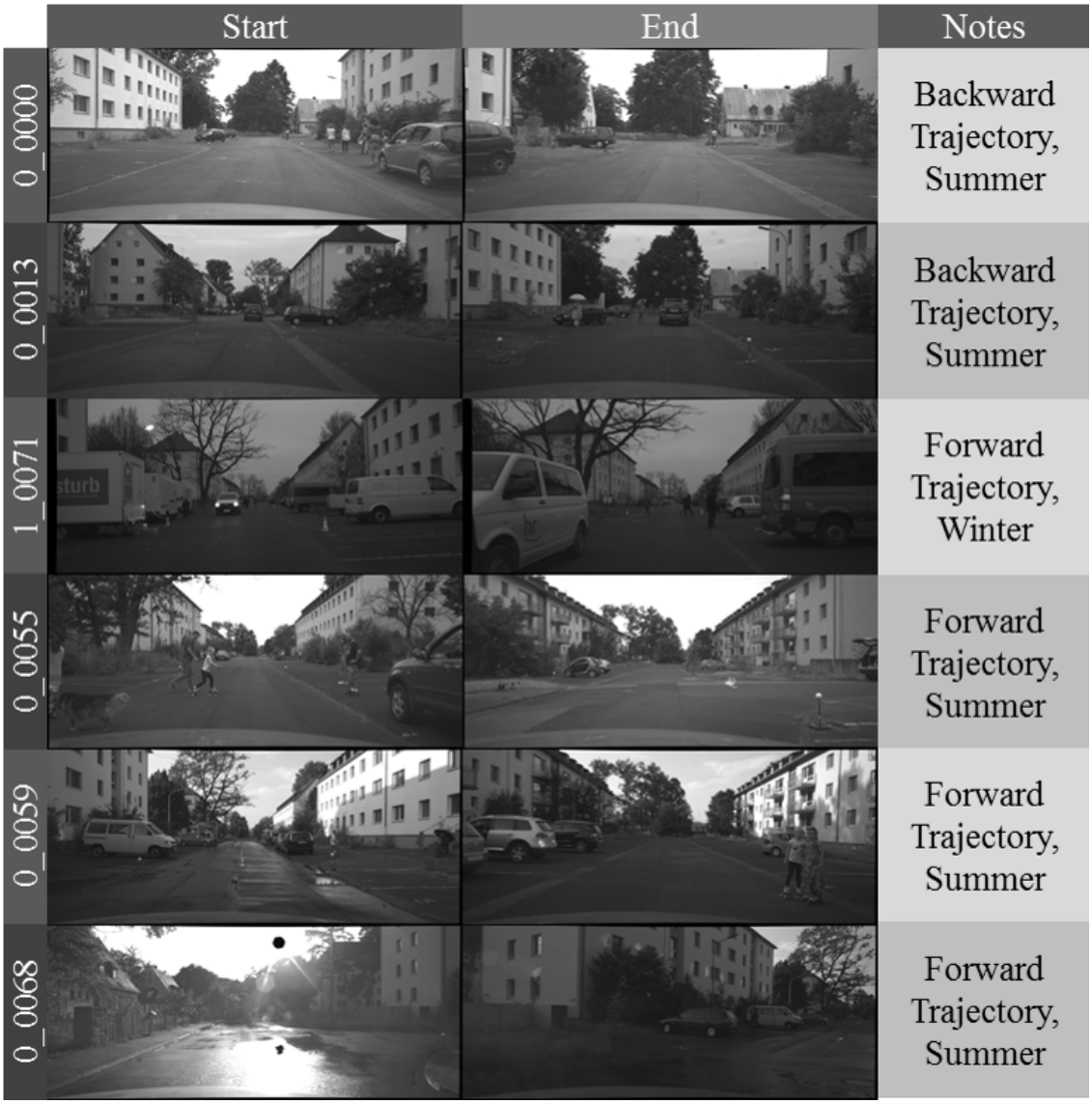


Figure 5.1: Illustration of the six sequences used for quantitative evaluation and their properties.

5. The subsequent sections 5.3, 5.4 and 5.5 evaluate the individual steps *feature matching*, *frame matching* and *outlier detection* by deriving initial requirements for the feature matcher, analyzing whether the frame matching meets these requirements and demonstrating the improvements of the outlier detection on the global solution. Finally, we discuss scene and sequence-related difficulties in section 5.6 on page 65.

5.1 Performance Evaluation

This section discusses the overall performance of the developed method and reasons on general limits. First, we show the applicability of the performance measure *Consistency* by assessing the related precision values σ. The performance overview will be given by reporting the consistency value distribution, thereby illustrating how many significant pose deviations are observed by applying our method to the test sequences. The remainder of this section discusses the limits of the method that are already apparent in the annotation result.

The following evaluation was made with respect to the test sequences that have been manually annotated to provide groundtruth, as has been explained in the introduction. For each frame pair, the information about consistency, precisions, number of available tracks etc. have been acquired and are saved as one datum. The data points presented here do not contain frame pairs where no annotation was possible. The failure of auto annotation can occur for instance due to a low track count of either reference or target frame or if the pose estimation and covariance analysis fails on that given frame pair. In this case, the method signifies to the user that auto-annotation failed. We do not consider these cases here as they would require manual intervention anyways.

Consistency Check We learned from the beginning of this chapter that using *consistency* as a performance measure brings along requirements on the related precision values σ, hence we gauge the precision value distribution in the following discussion.

We require the precision values to scale to the scene and that the precisions of groundtruth pose and auto-annotated pose exhibit similar scales. An overview of the precision scales can be gathered by examining their distributions. Translational and rotational precisions cannot be represented as one scalar because they have different units. The histograms shown in figure 5.2 on the facing page illustrate the respective distributions separately for translational (σ_t) and rotational (σ_r) precisions. First, we focus on the translational histograms. By comparing groundtruth and auto-annotated precisions we find that the precision distribution of groundtruth proves to be tighter with a tendency to smaller precision values. This is to be expected as human annotations are considered to be less prone to errors and therefore lead to less steep derivatives in the objective function, thus yielding smaller precision values. However, both precision distributions show a significant overlap as the following quartile ranges demonstrate. The consistency values are obtained by the translational

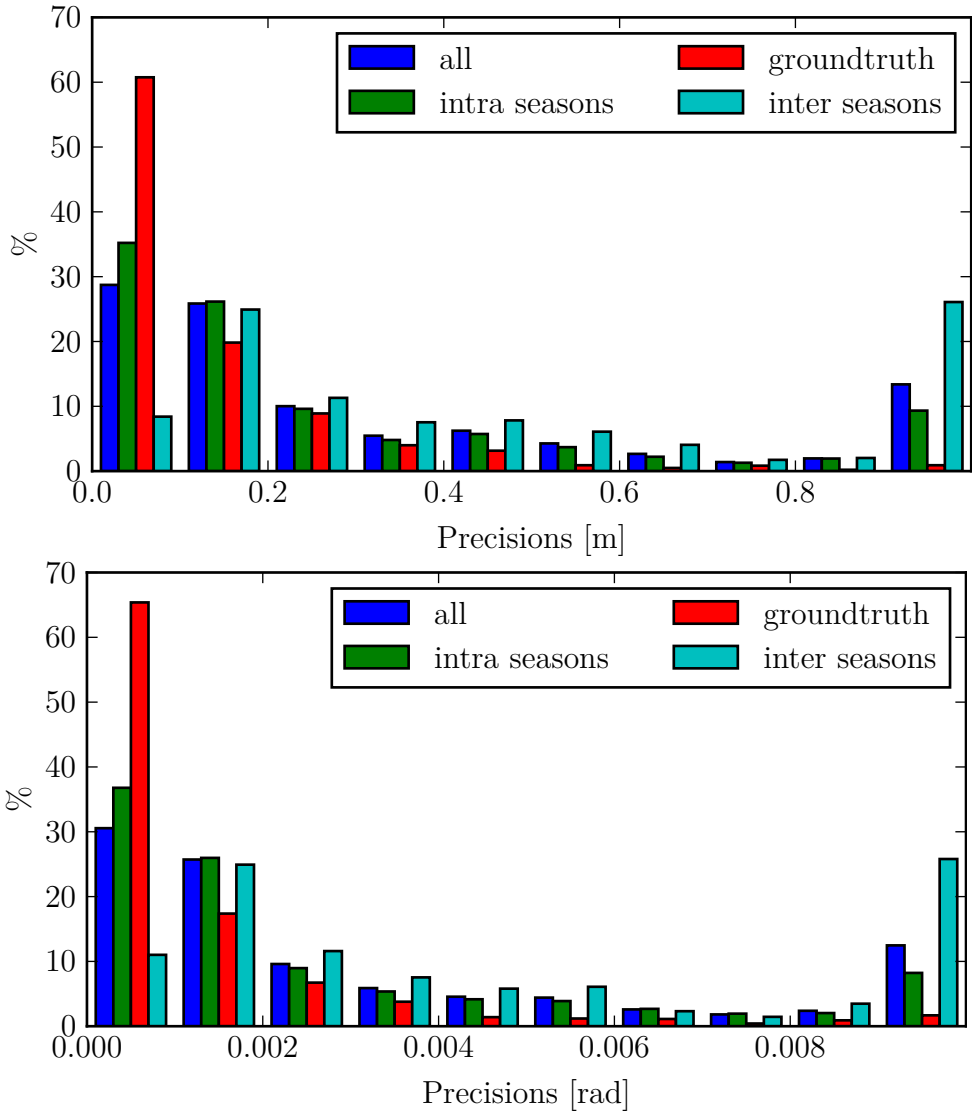


Figure 5.2: The Mahalanobis Distance, giving rise to consistency values, depends on the covariance matrix Σ. Consistency values, in our case, are only meaningful if precisions show a comparable scale because the covariance matrix is the sum of the camera pose covariances. We report the precisions as $\sqrt{tr(\Sigma)}$ of the manual annotation as well as automatic annotations. Rotational precisions are given in *rad* due to $\Delta\alpha \approx tan(\Delta\alpha) = \Delta r/r$ for small $\Delta\alpha$, allowing a quick estimation on point's error. Again, we split up the automatic results into inter- and intra-season sets. The top histogram shows sum of translational diagonal, the lower histogram shows sum of rotational diagonal entries of the covariance matrix Σ. Besides bins that containing low and high precision values we see a comparable distribution between manual and automatic annotations. Manual precision values tend to smaller values, precisions of automatic annotations show heavy tails. Tails of intra-season annotations are smaller, though. The precision distributions show a overlapping region, suggesting that the consistencies given in figure 5.5 on page 47 are comparable.

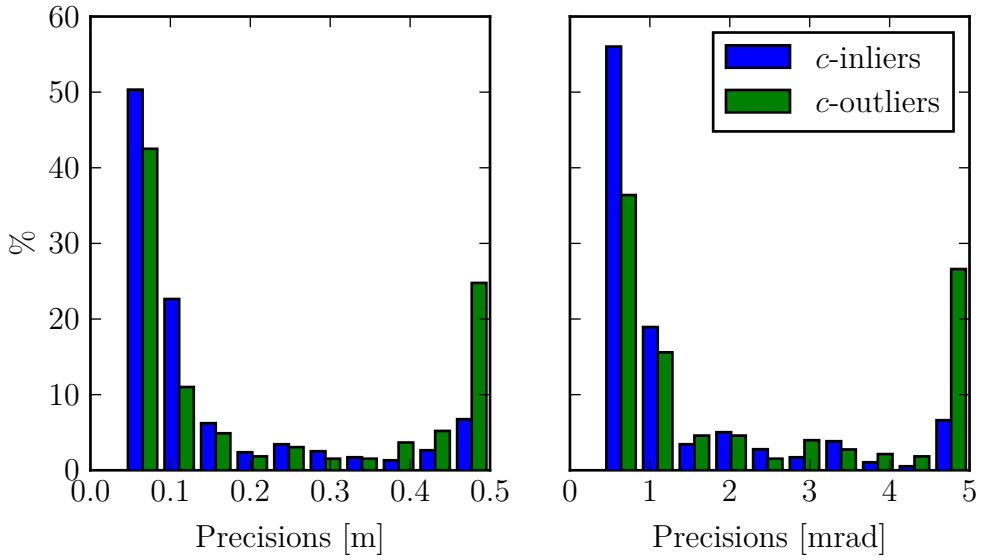


Figure 5.3: Precision distribution of intra-season inliers (consistencies $\leq 3c$) and outliers. The distribution of precisions is similar between in- and outlier, with the tendency to higher precision values within the set of outliers. 6% of the inliers have translational precision values $> 0.5m$ and 21% of the outliers respectively. This figure shows that the set of consistent annotations are not solely made up by poor precisions and that precision gives rise to differentiate between good and bad annotations.

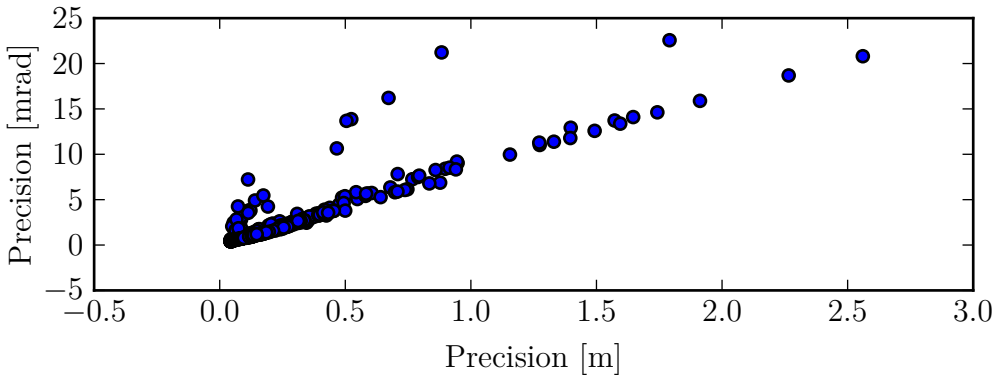


Figure 5.4: Correlation of translational and rotational precision values. Both precision components show a strong correlation. This suggest that it is sufficient to only take one of both components into consideration for further discussion.

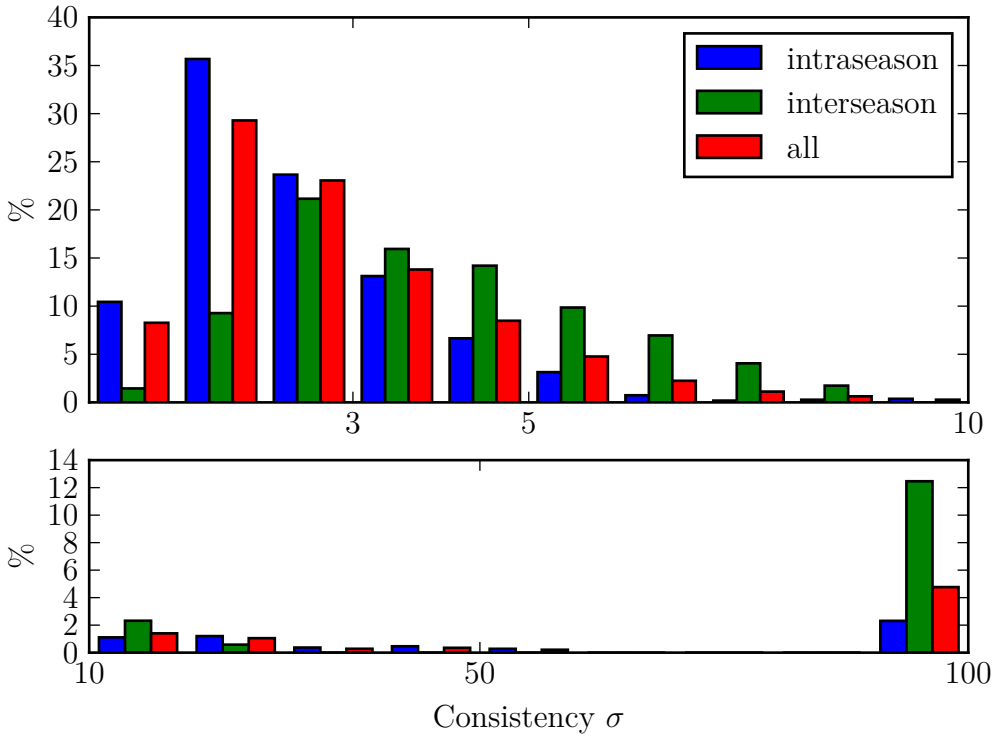


Figure 5.5: Evaluation of annotation quality, acquired by processing the whole pipeline on the test sequences. This histogram shows the distribution of pose consistencies which are obtained by calculating the *Mahalanobis Distance* between groundtruth and auto-annotation poses. Consistency values greater than $3c$ are considered to be inconsistent with the groundtruth pose. Bar height resembles the proportion to all samples of their group. Upper plot shows distribution near small consistency values, lower plot shows the distribution tail. The results are split into inter- and intra-season annotations, showing a shift to higher consistency values for inter-season annotations and a heavier tail as well. The lower plot indicates that most of annotations are made near the threshold of $3c$. All consistency values have been clipped to 100. Quantitatively, 70% of the intra-season annotations and 32% of the inter-season annotations fall inside the $3c$ range, in total consisting of 1082 and 345 samples respectively.

precisions of 0.14 m being within the 75 % percentile and 0.25 m being within the 75 % percentile for groundtruth and automatic annotations, respectively. This shows that the precisions of groundtruth and auto-annotated poses scale as expected. Assessing the scale of precisions with respect to the scene can also be done using quartiles. The majority of precision values are below 0.25 m. Considering that the selected 3D landmarks have uncertainties of a few centimeters and that many of them are co-planar we see that the translational pose uncertainties scale reasonably within the uncertainties of the scene geometry. The co-planarity can be seen in figure 5.1 on page 43. Here, most of the annotated landmarks are drawn from the house facades, which can be considered mathematical planes.

Next, we focus on rotational precisions. By comparing rotational with translational histograms of respective groundtruth and auto-annotated pose we see a notable similarity. We argued in the introduction that pose position and rotation correlate which we can then expect for precision values as well. In order to show that the translational and rotational precisions correlate, we plot rotational over translational precisions in figure 5.4 on page 46. In this figure, we see that the precision pairs indeed strongly correlate which proves our assumption and explains why the histograms look very similar. This result also suggests that it is sufficient to only take one of the precision components into account for further discussions.

We defined annotation inliers and outliers in terms of *consistency* as having consistency values lower or greater than $3c$, respectively. It is possible that inliers only occur together with very poor precision values whilst all outliers only occur given very small precisions. If this were the case, *consistency* could certainly not be used as a meaningful measure to distinguish between good and poor annotations. To asses this objection, we show the distribution of precision values conditioned on inliers and outliers in figure 5.3 on page 46. From this figure we learn that the precision distributions for inliers and outliers look similar and exhibit reasonable overlap. Furthermore, we see a tendency to smaller precision values for inliers. The similar precision distributions between inliers and outliers show that our initial concern does not hold. Instead, we see the opposite of our initial objection: Inliers are more credible as they show smaller consistency values *together* with smaller precisions.

The former discussion confirmed that different frame pairs imply different scales of uncertainty which needs to be taken into consideration for a fair quality measure. It has been shown that precision values are similar for automatically annotated poses and groundtruth poses, which in turn shows that classified inliers and outliers not only stem from mismatching precision values. In conclusion, we have proven that the *Mahalanobis Distance* and the notion of *consistency* provide an applicable annotation quality measure and that the use of such a more generalizing error measure is justified.

Annotation Quality Having proven the performance measure to work as expected, we can finally discuss the annotation quality by applying the developed method on the groundtruth-equipped test sequences.

We first give an overview of how well the method performs given the test sequences.

We show the distribution of all data points that were acquired using the test sequences and were reported to be successful annotations. The histograms can be seen in figure 5.5 on page 47, with the overall distribution being denoted by *all*.

We can see that 39 % of all sequence pair samples fall outside of a Mahalanobis distance range of $3c$, denoting false annotations. Generally, we do not expect our method to work seamlessly for all sequences and within all parts of a sequence due to the requirements that have been imposed on the sequence dataset as explained in section 2.1 on page 13. The sequences were recorded in order to serve as challenging vision tasks and requiring the annotation method to work perfectly would render the sequence set pointless in some sense. Instead, the proposed method was chosen to be a semi-automatic method expecting the user to revise the results and correct them in case of a failure. This suggests that the presented outlier rate is reasonable and acceptable.

However, it would be beneficial to find limits of the method that can be used a priori and a posteriori to detect whether annotations are good or bad, thereby directing the user to parts of annotations that need more attention. Hence, in the remainder of this chapter, we aim to assess the reasons behind the outliers and detect limitations that are inherent to the annotation method.

We have reasoned before that the proposed method requires visually similar sequences. From visual inspection of the sequence set (fig. 5.1) we can see striking differences between different seasons. In order to show that different seasons have an impact on the annotation performance, we separate the combined consistency value distribution into intra- and inter-season pairs. The separated distributions are featured in figure 5.5 on page 47.

By comparing inter- and intra-season annotations, we immediately see a correlation of poor annotation quality with inter-season annotations. As it turns out, many errors stem from the auto-annotation of inter-season pairs with 32% of the poses being inconsistent in this context. This can also be understood by looking at figure 5.1 on page 43. Leaves fall from the tree, thus revealing objects behind the trees. This has an impact on the visual correlation and influences the sensitivity of the coarse alignment. We hence identify inter-season annotation as a serious shortcoming of the proposed method, with the latter clearly lacking applicability here. We therefore only consider intra-season pairs in the following discussion, if not denoted differently.

We have analyzed the annotation quality obtained by applying the proposed method on the test sequences and shown that the method works well given the challenges provided by the test sequence set. Nevertheless, there remain 30 % outliers in the intra-seasons pairs. We aim to further understand the reasons behind these outliers with respect to the capabilities of the annotation method in the following sections.

5.2 Limits

This section is dedicated to finding general limits of the developed annotation method. We will use the obtained results as a posteriori features to indicate good and bad annotations to the user.

In order to find the general limitations of the annotation method, we recall what the method does: We register one frame with respect to another, requiring feature points (2D) in both reference and target frame. Low annotation counts imply higher numbers of possible annotation constellations. Hence, the number of features and the respective number of made annotations need to be taken into account. Secondly, we find initial pose estimates by determining the visually most similar frame pairs for reference and target frames. Visual similarity depends on camera distance as frame pairs that are too distant to each other do not have much in common anymore. Hence, it is probable that these factors influence the annotation results. Therefore, we want to understand whether the consistency values c exhibit a clustering with respect to reference camera distance ΔT and the number of made annotations N. Corresponding data is shown in the scatterplot in figure 5.6 on the facing page.

In this figure, we see that inconsistent points ($c > 3c$) are uniformly distributed within the data points. Consistent samples, on the other hand, can be found at low numbers of annotations as well as large camera distances between reference and target frames. Contrary to our assumption, we see that we cannot find evident indications in the consistency distributions of camera pose and number of annotated tracks.

We can understand this from the following reasoning: Camera distances for the presented datum points were the result of coarse matching. Hence, the distances reported here heavily depend on the performance of the coarse matching step. Therefore, these results can not be used particular well for a systematic examination as we don't have enough control over the results. We therefore pose the same question in section 5.3 on page 54 again using a systematic variation on camera distances.

Another parameter which we can learn from are the precision values as they are an indicator for the quality of the pose estimation. It is reasonable that for comparable annotations these precisions should match as well. We want to compare the uncertainty levels of the reference annotation and the target annotation. If the target precision deviates much from the reference, then it is likely that something has gone wrong. For comparison, the distributions of the ratios between reference precisions σ_t^R and target precisions σ_t^T have been split up into inliers and outliers and are shown in figure 5.7 on page 52.

Looking at both histograms, we see that the precision ratios behave as expected. Clearly, most of the ratios lie between 0.5 and 2.0 with a peak around 1.0. Looking at the inliers separately, we see that 50% of the values lie between a ratio of 0.98 and 1.25 denoting the first and third quartile within the inlier distribution. When analyzing the same distribution for outliers, we see similar values lying between 0.98 and 1.64, respectively. Values greater than 1.0 denote values where the target precision is greater, denoting a worse fit for the target annotations which sounds

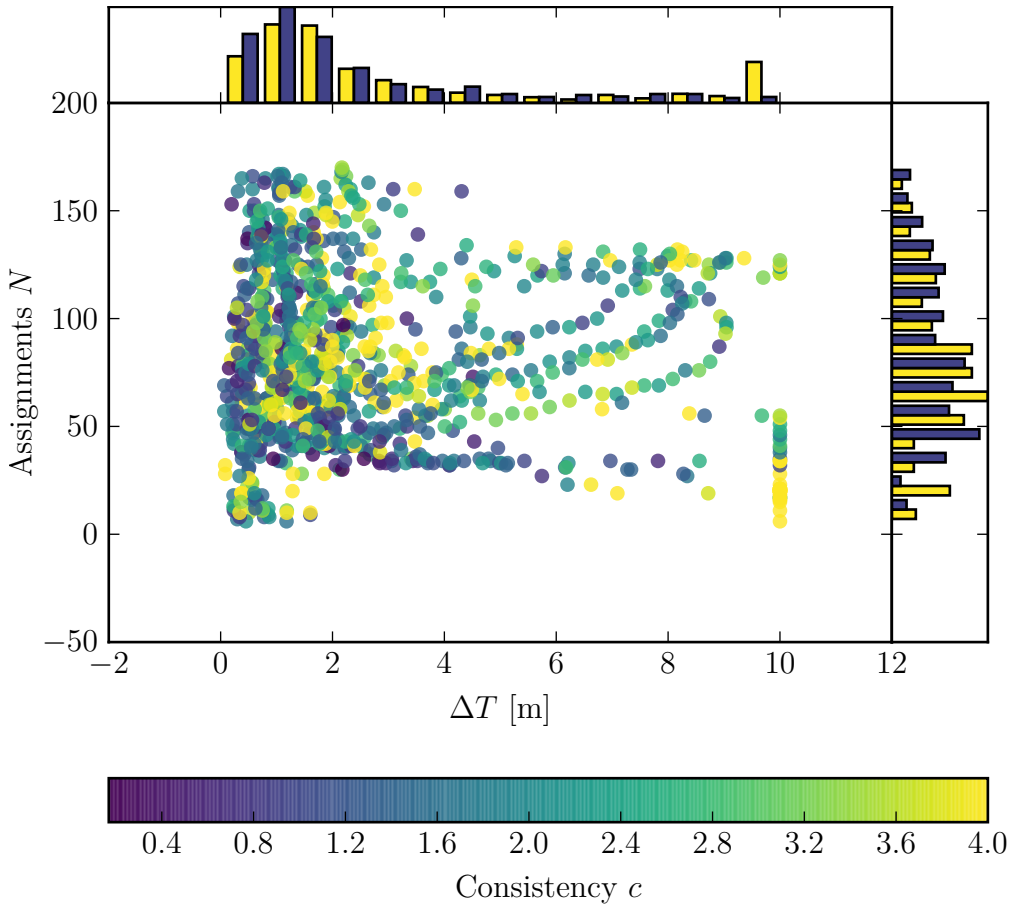


Figure 5.6: In order to understand the cause of outliers in intra-season annotations, we dissect annotation quality with respect to extrinsic parameters *reference camera distance* ΔT and *number of annotations*. The consistency is color coded and has been clipped at 4 for better visualization. The marginalization of outliers (consistency $> 3c$) is shown in the respective histograms. Note the parabolic slopes as well as the vertical line originating from clipping camera distances No true coherence can be found here, as bad and good consistency values are scattered uniformly within their distribution.

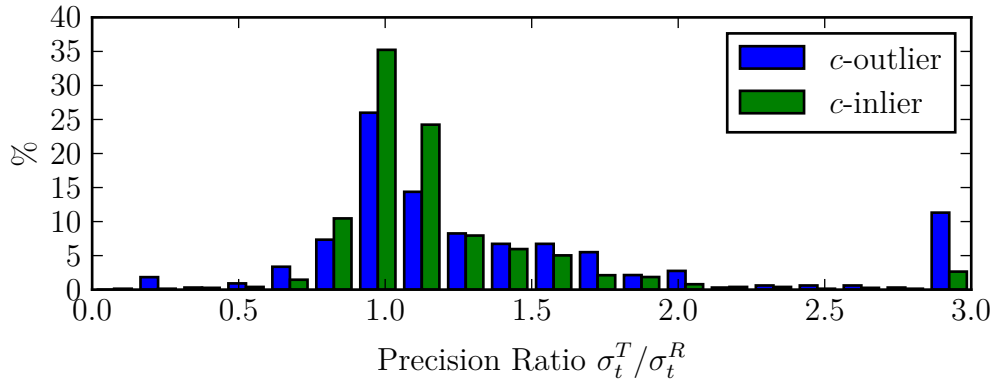


Figure 5.7: Assessing the useability to discriminate outliers using absolute precisions values. The ratio between reference and annotation pose precision is shown on the abscissa. The histogram has been split up for in- and outliers. One can see that the majority of pairs have ratios between 0.5 and 2. Outliers show higher values for larger ratios. However, in- and outlier show similar distributions giving no hint on discrimination but showing a correlation between reference and target precisions.

reasonable. The figure proves our assumption about similar precision values between reference and target poses.

Comparing both distributions, we see that they extend similarly. Thus, classifying outliers by sole use of this distribution is not possible without also cutting away inliers. This concludes that precision values between target and reference correlate but that the precisions ratio provides no way of classifying outliers.

We continue the examination of the precision values. In the introduction, we argued that the number of annotations N and the reference camera distance ΔT also should have an influence on the precisions which we aim to analyze now. We report the precision values σ_t of each annotation pair in a scatterplot with respect to N and ΔT, similar to the former discussion about consistency values.

We first focus on the clustering depicted in the upper scatterplot. Precisions σ_t comparable to groundtruth precisions (≤ 0.1m) extend up to the clipping distance of 10m. This shows that the camera distance ΔT seems to have a negligible impact on the precision values. Looking at the number of assignments N, we see that the precision increases gradually with decreasing N. The 25% percentile for precision values equal or better to groundtruth precisions lies at a number of 60 assignments, thus indicating a clear dependency of the precision values σ_t on N. We can understand the two observations as follows. The impact of 3D positional errors on the reprojection error decreases with increasing distance to the points. Normal distributed uncertainties can be reduced by increasing the amount of samples. This suggests that the dominating error process on the reprojection error is the tracker positional noise with negligible influence of the 3D positions. This complies with our observations made.

Having identified the amount of annotations N as being the dominating parameter

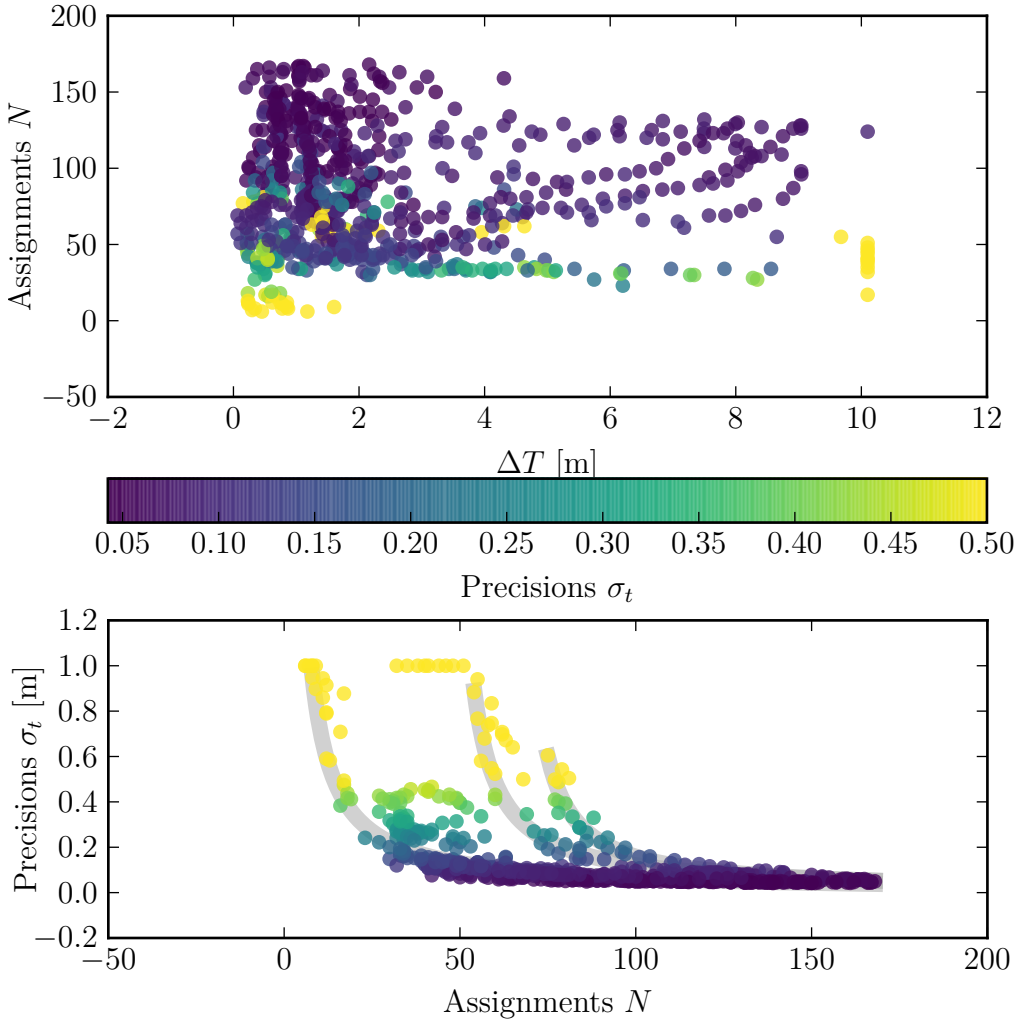


Figure 5.8: Influence of number of assignments N and camera distances ΔT on the precision values. The upper plot shows clustering of precision values. The points are color coded with respect to precision values as shown in the colorbar. Camera distances have been clipped to 10 m, leading to the vertical line on the right. The lower plot shows dependency of precision values σ_t with respect to number of assignments N. Three branches are evolving into which functions of the form $s_0/\sqrt{N+1-N_0}$ have been fitted.

on precision values, we further dissect this dependency by showing the precision values σ_t plotted over number of assignments N which can be seen in the lower plot of figure 5.8 on the previous page. The most striking feature of this plot is the clear convergence to small precision values with high numbers of annotations N, which we conjectured from the upper plot already. Despite this fact, we see a few branches evolving as reciprocal curves illustrated by the supporting lines. These branches have the same cause as the parabolic slopes in figure 5.6 on page 51. In the context of uncertainties the reciprocal curves stem from taking the mean of random variables, thereby scaling uncertainties $u \propto 1/\sqrt{n}$ with the number of samples n. In fact, the lines are fitted to each branch using following equation.

$$u(N) = \frac{u_0}{\sqrt{N + 1 - N_0}} \tag{5.1}$$

According to the figure, the reciprocal branches indeed obey the given proportionality up to a shift N_0, which is known to be valid for normal distributed errors. This indicates the validity of the assumption of normal distributed errors in the introduction of this chapter.

In the former discussion, we have identified the number of annotations N to have a significant impact on the precision values. Furthermore, we have shown coherence to a proportionality that we know is valid for normal distributed errors, indicating that the camera pose uncertainties indeed stem from normal distributed errors.

In this whole section, we have found a justification for the chosen performance measure *consistency* by showing that this relative measure shows no significant correlation with precisions, denoting that inliers and outliers classified via *consistency* is a true classification. On top of this, we identified a further indication that the pose uncertainties are normal distributed by showing a strong coherence to the number of assignments N.

Finally, we gave an overview of general limits of the annotation method but were not able to find satisfying and clearly inherent restrictions of the proposed method. Up to now, we only examined the annotation method as a whole, hence considering the system as a black-box. This black-box consists of three steps that each have an impact on the overall performance. In the following sections, we will analyze the three steps *pose refinement*, *pose estimation* and *outlier detection* in section 5.3, 5.4 and 5.5 individually.

5.3 Feature Matching

In this section, we examine the camera pose refinement step separately. This step is the central step within the whole pipeline. We aim at finding its limits and requirements. These limits and the isolated performance analysis can later be used to discuss the effects of *coarse alignment* and *outlier detection*.

In the former discussion, we analyzed the whole pipeline. Hence, the choice of reference and target frames was determined by coarse alignment. We now want to

impose systematically chosen reference/target frame pairs to the *pose refinement* pipeline, thereby retaining full control over the similarity and camera distances of both reference and target frame.

This experiment has been conducted by picking a small subset of frames from all sequences and using them as reference. A datum point for each of the frames is acquired by stepping away from the reference frame and thereby systematically recording the frame difference, camera distance and annotation quality. For both forward and backward direction, we use 20 steps with a step-width of 15 frames. We systematically step away from a frame starting with complete visual similarity. This effectively requires the initial frame distance to be 0 m. We can only ensure this by choosing frame pairs from the same sequences. Hence, the choice of frames has been limited to intra-sequence pairs.

We first give an overview by showing the annotation results with respect to the scene, thereby also illustrating the scene coverage of reference frames. The reference frames have been chosen to cover different aspect of the scene. This includes (a) the curve (b) frames within two houses and (c) frames in between two columns. Situation (a) includes camera motion with rotation , (b) includes highly repetitive structures and co-planarity of landmarks and (c) shows frames with mixed landmarks. This selection covers different properties of the scene, thereby giving a good cross-section of the difficulties. The distribution of (x, y) positions of the camera pose with color-coded consistency c are shown in figure 5.9 on the next page.

Qualitatively, we can see that the annotation results exhibit good consistency values if the target frame is still near to the reference frame. This shows the general applicability of the refinement step but suggests that it needs a good prior to the reference scene in order to work properly.

Next, we assess this observations quantitatively. This discussion has been postponed in section 5.2 on page 50 where we were interested in the clustering of consistency values c with respect to number of assignments N and the camera distance ΔT, but were unable to find a significant correlation. Using the systematically acquired dataset, we construct the same scatterplot again by showing the color-coded consistency plotted with respect to ΔT and N. One can reason that the number of false annotations also affects the annotation quality c, hence we also plot the error rate against the number of annotations N in a further plot. The error rate is the number of false annotations divided by the total number of annotations. Both plots can be seen in figure 5.10 on page 57.

Focusing on the top plot, we see a clear dependency of consistency c with respect on the camera distance ΔT. Here, we see that good consistencies are only obtainable by camera distances of less than 3.2 m within the 75% percentile and 5.5 m for the 95% percentile. This agrees with figure 5.6 on page 51 which shows a similar distribution. For the number of assignments N we can find that good pose consistencies can be achieved for low as well as high N. This implies a less pronounced dependency on consistencies c. A lower limit can be identified to be 49 annotations considering datum points above the 25% percentile. Now looking on the error rate plot below shows that it is necessary to maintain an error rate of less than 20% to achieve good

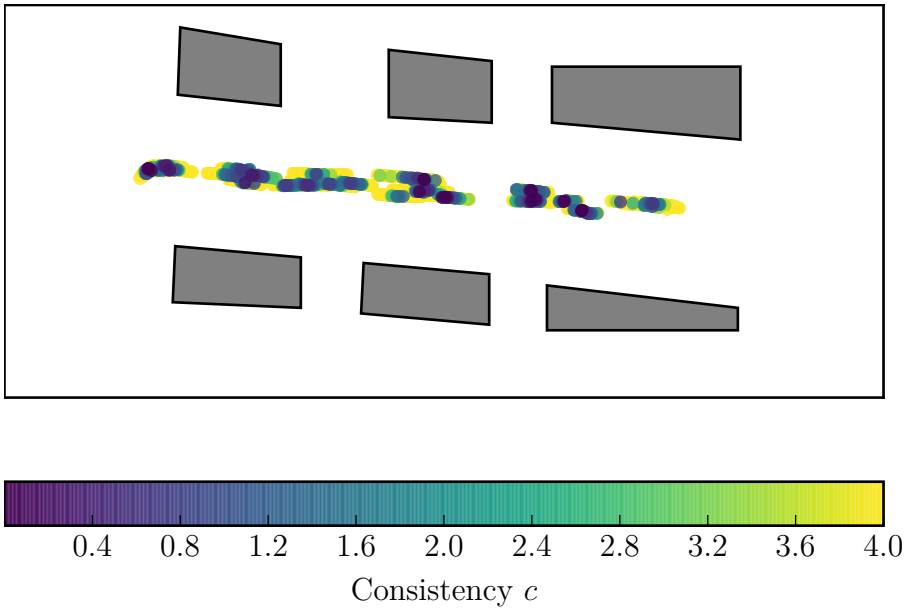


Figure 5.9: Bird's eye perspective on the annotation results obtained by stepping away from reference frame. The consistency is color coded. Consistency was calculated between groundtruth pose of target frame and the annotation made for the target frame. One can see good consistencies near the reference frame, evenly dropping to bad consistencies before and after the reference.

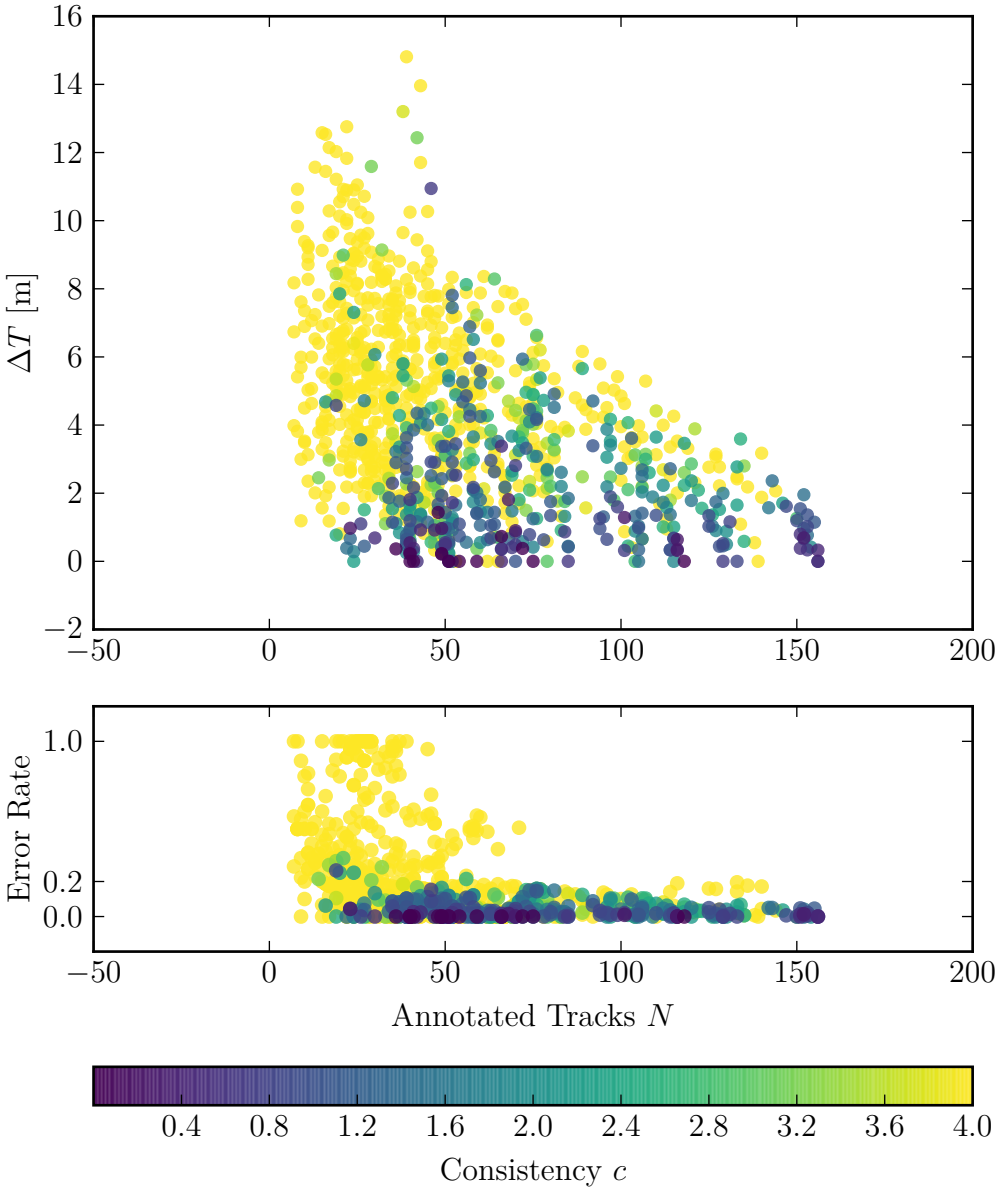


Figure 5.10: Systematic approach on assignment quality. Both plots show the distribution of pose consistencies obtained by fixing a reference frame and annotating target frames evenly spaced steps away from the reference, using the same sequence. Consistency is color coded. The top plot shows the distribution given the camera distances ΔT, bottom plot shows the error rate. One can see that good consistency values are only obtained for number of assignments larger than 40 and camera distances not more than 5 m away. Satisfying annotation qualities can only be achieved with error rates less than 20 %. This plot shows practical limits on camera distances between target and reference frames and the number of required annotatable tracks.

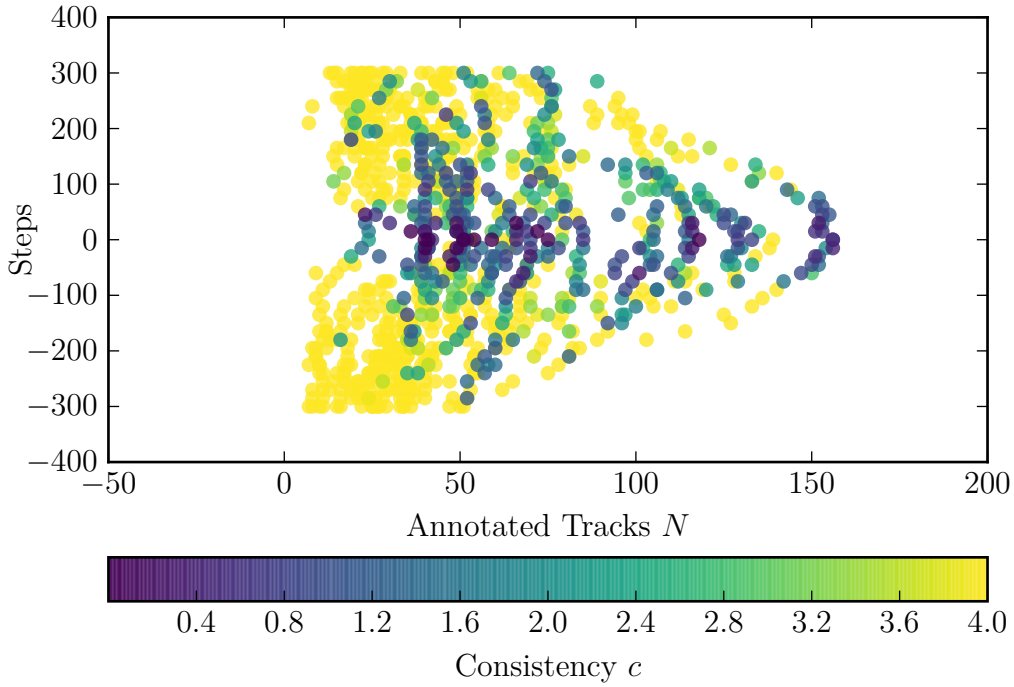


Figure 5.11: Extends figure 5.10 on the previous page by the dependency of inter-frame-stepping.

annotation results. We see a region below a count of 50 annotations where the error rate increases notably. This complies with the limit on annotation count we have identified in the plot above.

We understood that annotation quality depends on frame camera distances. This entity is not known a priori as groundtruth would be required. Hence, this cannot be used directly as an indicator for the quality of annotation. However, we can use this information by finding situations where the frame matching recurringly shows higher distances than 3.2 m. If regions in the scene show such recurring higher distances then the annotation will be likely to be poor in this case. Hence, the further discussion about this will be postponed to section 5.4 and section 5.6.

Another parameter that is relevant for *frame matching* is the inter-frame stepping. This parameter defines how many frames should be ignored in between two frames that are to be annotated. We investigate how many frames the *frame matching* step can be away from the true frame to still work. To this end, we report the color-coded consistency with respect to the frame number difference and the number of annotations N. The resulting plot is shown in figure 5.11.

We see that the frame stepping distance has a coherence with the consistency c. At first glance, the plot suggests that the annotation is symmetric to the frame difference. However, the 50% percentiles for frame differences higher than 0 and lower than 0 are found to be -45 and 67, respectively, where the choice of percentiles effectively resembles the interquartile range of the whole dataset. We see an asymmetry for annotations before and after the reference frame. This asymmetry can be understood

from the reference frames that have been placed at the boundary of available groundtruth. In these positions, we cannot find annotations in the backward direction due to missing tracks. Hence, these positions will not contribute annotations behind the reference frame, thereby reducing the number of annotations in the negative branch. From the discussion about frame differences we know that we should not be more than 50 frames away from the true frame.

5.4 Frame Matching

In the former section, we have identified general limits on camera distance or number of assignments in order for the feature matching to work properly. Our method consists of Frame Matching followed by the frame wise feature matching. We now aim to verify that the frame similarity requirements necessary for the feature matching are provided by the frame matching. In addition, we aim to show the useability of the developed inference-based method. The diagonal path through the similarity map is the baseline here. In this context, we show the improvement of the inference-based frame matching over the naive *diagonal* and *unary max* alignment as mentioned in section 4.3 on page 33.

The underlying quality measure of this chapter is as follows. We intend to compare how well the frame matching works. Goal of the implemented frame matching is to find the visually best matching reference frame for each target frame. We have seen that visually similar frames possess a similar camera pose. Therefore, as quality measure, we define the camera distances ΔT. These distances are obtained using the test sequence groundtruth poses. For each assignment (T_j, R_i), we record the corresponding camera distance ΔT. In principal, we would have to compare the rotational part as well. However, most of the camera trajectory is straight movement. Here, the rotational components don not change. Only when driving the curve, we encounter bigger changes in the rotation vectors. We will find a justification in section 5.6 on page 65 such that we can neglect the rotational part for the comparison of the methods.

We first focus on the general improvement of the graph inference step compared *diagonal* and *unary max*. For all test sequences, we used these three methods and compared their correspondences in terms of the camera distance. The histogram of all camera distances is shown in the top plot of figure 5.12 on the following page. We see that the *diagonal* approach exhibits an initial distance of over 13m for the 75% percentile. The unary term alone produces better results with 5m in the same percentile. However, we see a heavy tail similar to *diagonal*. For graph inference, we find that 75% of the values are smaller than 3m whereas the 95% percentile can be found at 10m. These observations shows that the graph inference produces matchings with a less pronounced tail and proves the higher usability with respect to the former methods.

We now compare these results to the requirement for the feature matching that we have identified in section 5.3 on page 54. From figure 5.10 on page 57, we learned

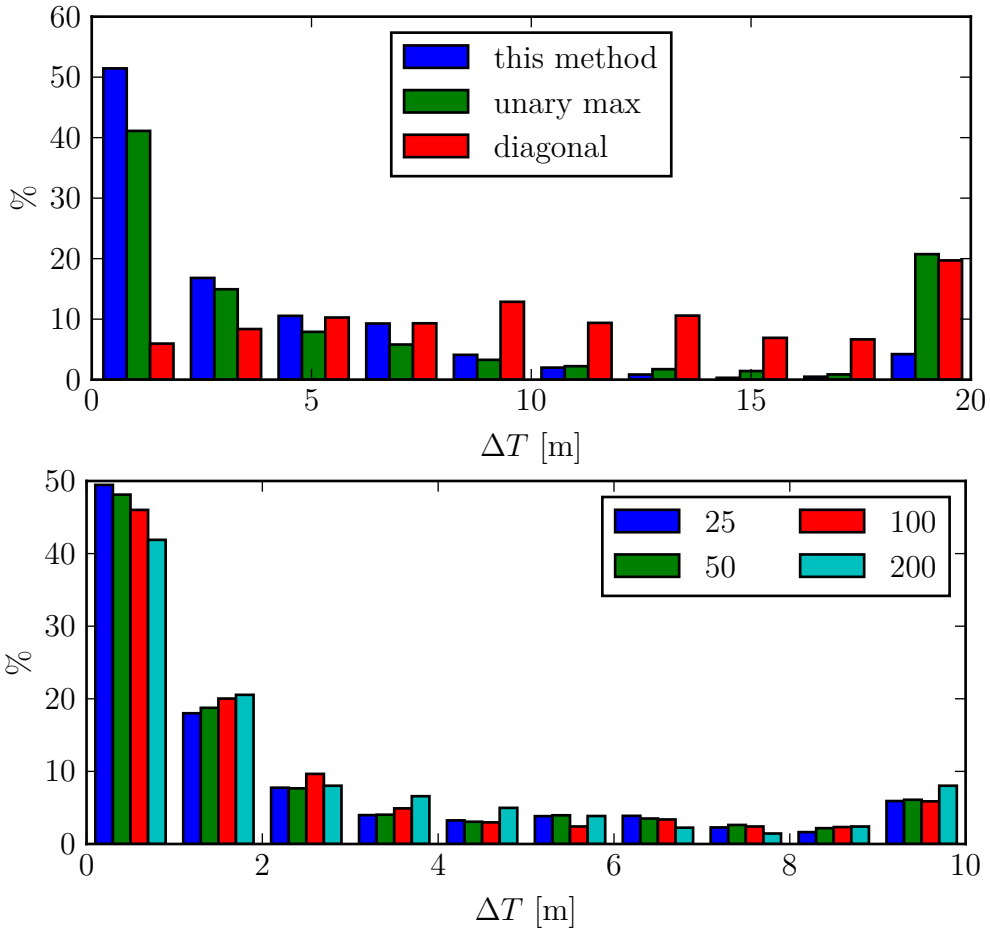


Figure 5.12: Error distribution of coarse alignment step as described in section 4.3 on page 33. The error is computed using the difference of camera position of true and suggested pose. The upper plot shows the comparison between our method, unary max and diagonal path. The narrow error distribution of our method significantly drops to small values for deviations higher than 10m. The naive diagonal approach shows a uniform distribution within the range of 20m. Compared to *unary max*, our method is much less heavy-tailed in error distribution. The lower plot shows the influence of inter-frame stepping of the coarse alignment. It can be seen that the errors for different inter-frame steppings have similar distributions.

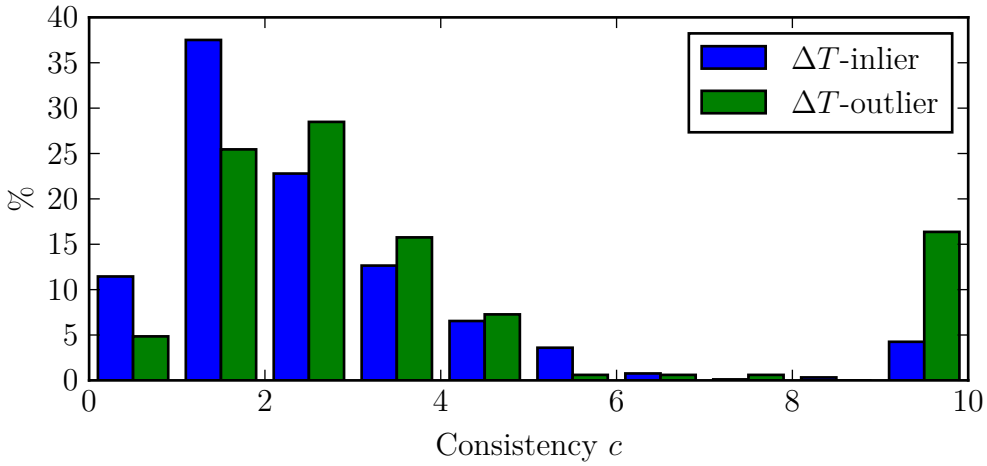


Figure 5.13: Consistency distribution for frame matching conditioned on inliers $\Delta T \leq$ 5m and outliers.

that the refinement step ideally requires camera distances of less than 5 m to work properly. This is not completely guaranteed by the coarse alignment as approximately 15 % of all frame pairs exhibit distances greater than that.

This suggests to find whether the higher frame distances truly have an impact on the annotation results. We therefore condition the samples on inliers $\Delta T \leq 5$ m and outliers and report the consistency distributions of both subsets in figure 5.13.

Here, we see for inliers that the 75%-percentile lies at 2m and for the 95% percentile it lies at 4m. For the outliers we find the respective percentiles at 11m and 33m. This clearly denotes a shift towards higher and inconsistent auto-annotations which implies that the reasons for the annotation outliers presented in section 5.1 on page 44 can also be found in insufficient pose estimations.

We have shown that a large camera distance in the frame-matching step correlates with poor annotation quality. We now assess whether we can use this information in order to warn the user about poor annotations. Looking on the outlier distribution, we see that 58% of the samples are inliers in terms of consistency. This shows that we would throw away many useable annotation pairs if were to omit all frame matchings with camera distance larger than 5m. In addition , the camera distances are not known a priori. To use this information we have to find locations in the scene or sequences where we consistently find poor frame-matching results that meet these requirements. Concluding, the found correlation alone is not sufficient to detect outliers but we have to resort to a more scene-agnostic examination, which we do in section 5.6 on page 65.

In the performance overview in section 5.1 on page 44 we found that the inter-season annotations in general yield poor performance and reasoned that the frame matching step fails at this point. In order to prove this assumption, we show the camera distance distributions ΔT for intra-season and inter-season annotations together with two example similarity maps in figure 5.14 on the next page.

From this figure, we learn that frame matching in different seasons has a significant

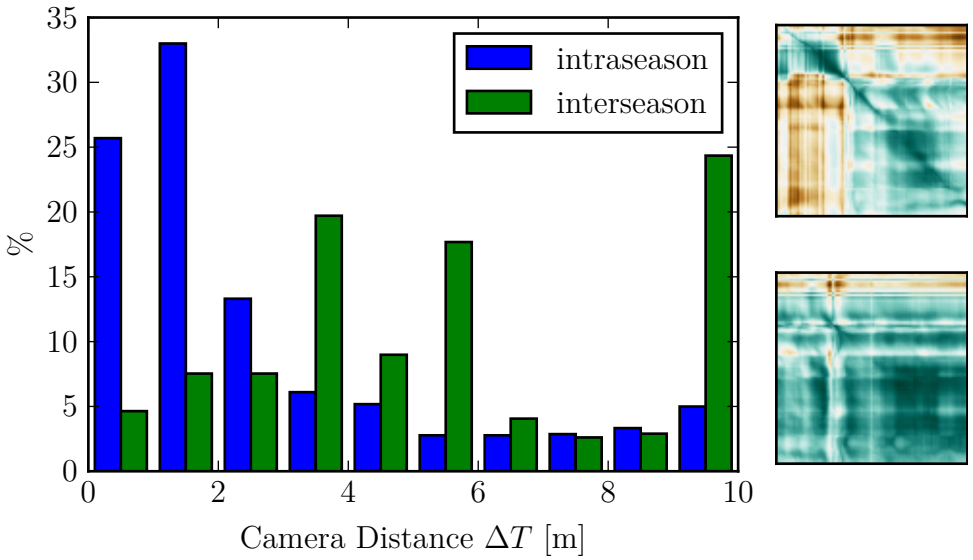


Figure 5.14: Camera distance ΔT of frame matching conditioned on intra and inter-season sequence pairs. We see a significant shift towards large camera distances for inter-season matching. Two examples for similarity maps are shown on the left hand side. The upper plot corresponds to summer/summer pair, lower plot corresponds to winter/summer pair. A path with higher contrast can be seen in the summer/summer similarity map.

effect on the matching quality. We see a drastic shift towards higher camera distances for inter-season annotations. This shift explains why the annotation quality for inter-season sequence pairs generally shows poorer results. By looking on the example similarity maps we see that for a summer/winter pair the contrast is far reduced. It is hard to find a viable track at all. Hence, the proposed method should only be applied to sequences from the same season as was reasoned before.

Another feature of the frame matching is the frame subsampling as explained in section 4.3 on page 33. There, we reasoned that many of the frames provide redundancies and concluded that it is sufficient to skip frames. The more frames are skipped, the more unlikely it is to find a good match. We want to assess the influence of frame subsampling on the coarse alignment quality in the following discussion. To this end, we apply frame matching to a list of subsamples [25, 50, 100, 200]. The number corresponds to how many frames are skipped between to consecutive frames. The steppings were chosen to lie around the limiting case of around 100 frames identified in the former section. The resulting error distributions can be found in the lower plot in figure 5.12 on page 60.

Comparing the different results, we see a slight tendency to higher frame distances for bigger step sizes. All in all, the error distributions look similar for all step sizes. Considering again the assumed velocity of $8\,\frac{\mathrm{m}}{\mathrm{s}}$, we understand that the frame distance for a stepping of 25 frames should be at maximum 0.5 m and 4 m for a stepping of 200. Half the frame distance assumed by driving speed $d = v/f$ is considered: if a frame deviates further away than half of d, the matcher is expected to find the next or previous frame instead. Given these observations, we find that the resolution of coarse matching does not depend so much on the frame stepping but is dominated by the limited resolution of the similarity map and the principal frame distances between different sequences.

From figure 5.11 on page 58, we learned that we see a limit when employing frame steppings above 100 frames. Since precision is not getting drastically better by choosing smaller step sizes, this suggests choosing a stepwidth of 50 frames as a good compromise between processing speed and resolution.

5.5 Outlier Detection

In section 4.4 on page 38, a postprocessing step has been proposed to correct outliers. The pose consistencies that have been presented in section 5.1 on page 44 were calculated using annotations that result after the postprocessing step. Here, we assess the impact of postprocessing on the quality of annotations.

To this end, we calculate the pose consistencies in the following experiments just as in section 5.1 on page 44. This time, however, we omit the postprocessing and present consistencies that are obtained only by taking the annotations that were found by graph-matching. The annotation pipeline is again processed for the presented test sequences. The resulting consistency distribution is shown in figure 5.15 on the following page.

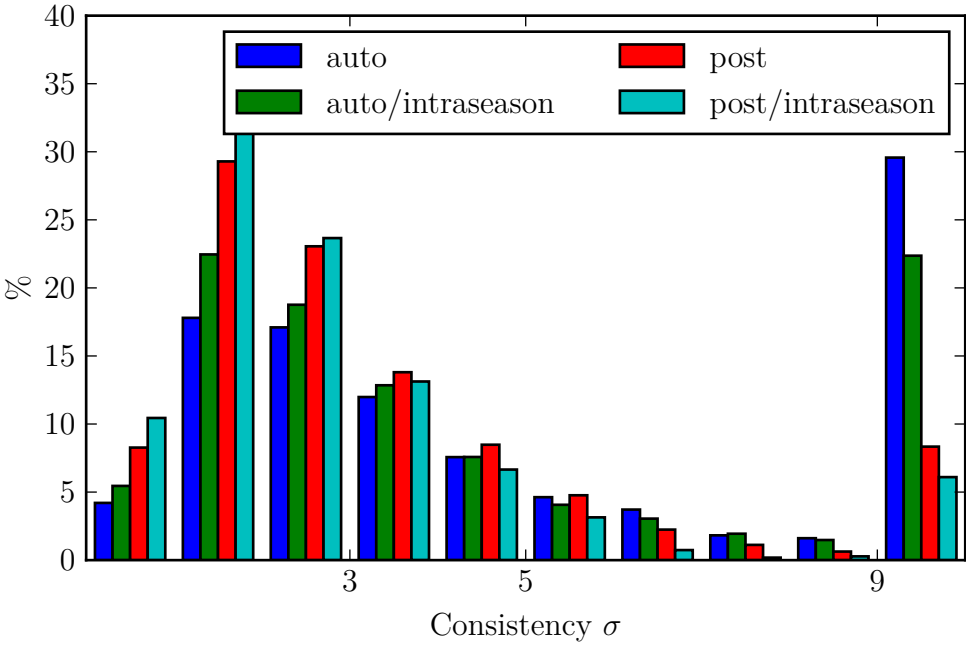


Figure 5.15: Comparison of consistency distribution between frame-wise annotations and annotations obtained after postprocessing step. Entries labeled with *post* correspond to the values shown in figure 5.5 on page 47, values with *auto* correspond to intra-frame annotation results. Results obtained by postprocessing, denoted by *post*, show a tendency to smaller consistency values. This can clearly be seen for values larger than 9σ, as the number of occurrences drops from over 20% to little more than 5%. This figure shows an impact of the postprocessing result on the overall performance.

Comparing the obtained consistencies, we find a decrease in consistency values using the proposed postprocessing method. The outlier rate drops from 61% to 39% in the overall performance and from 53% to 30% for visually similar sequence pairs. Thus, postprocessing corrects approximately 1/5th of the annotations. This result shows that the postprocessing step is not just a refinement to the results but turns out to be a vital step for an unsupervised pipeline workflow.

Using this observation, we look at some results retrospectively in order to understand them better. In the evaluation of the isolated feature matching step, we found a limit of 3.0m for initial camera distances to work properly. This partially contradicts figure 5.6 on page 51 where we found the parabolic slopes. These slopes consistently denote good annotation qualities which suggests that high-distant annotations are not solely achieved though intra-frame annotations but are supported through a *knock-on effect* of the postprocessing.

5.6 Scene & Sequence Related Difficulties

From the former sections, we learned about general limits to the application of the proposed annotation method. In this context, we examined the influence of common parameters like camera distance, number of assignments and precision on the consistency values. We furthermore examined the individual step *feature matching* and compared the identified requirements to the prior step *frame matching*. We found that the requirements for feature matching are not completely met. We showed for all of these indications that they have an effect on the annotation quality. However, all indicators were not feasible to clearly tell apart good from bad annotations, as there always was a significant overlap. In the following we want to find whether combinations of the former discussed indicators can be used as a a priori criterion for outlier classification.

Generally, we learned that we have to expect different quality of annotations for different sequences. One reason for this is found to be scene-related constraints. Not every part of the scene has a very dense point cloud. Some parts also show occlusions of potential landmark candidates. These scene-related difficulties are the subject of the following discussion.

We expect scene-related effects to occur locally and coherently. Hence we plot the consistencies c with respect to their spatial position in the scene and marginalize the distribution on in- and outliers. In addition to consistency, we report the precision values again. We also intend to understand the impact of frame matching. Hence, we surround the consistency scatter points with gray circles whose size is proportional to the reference camera distance. Interesting points are labeled. The resulting plots can be seen in figure 5.16 on the following page. To give a better insight on the local properties of the scene, we also show sample frames for the corresponding points of interest in figure 5.17 on page 69. Besides consistency, we examine the number of annotations and the error rate which can be seen in figure 5.18 on page 70.

We first focus on the consistency plot. The most striking mode in the distribution

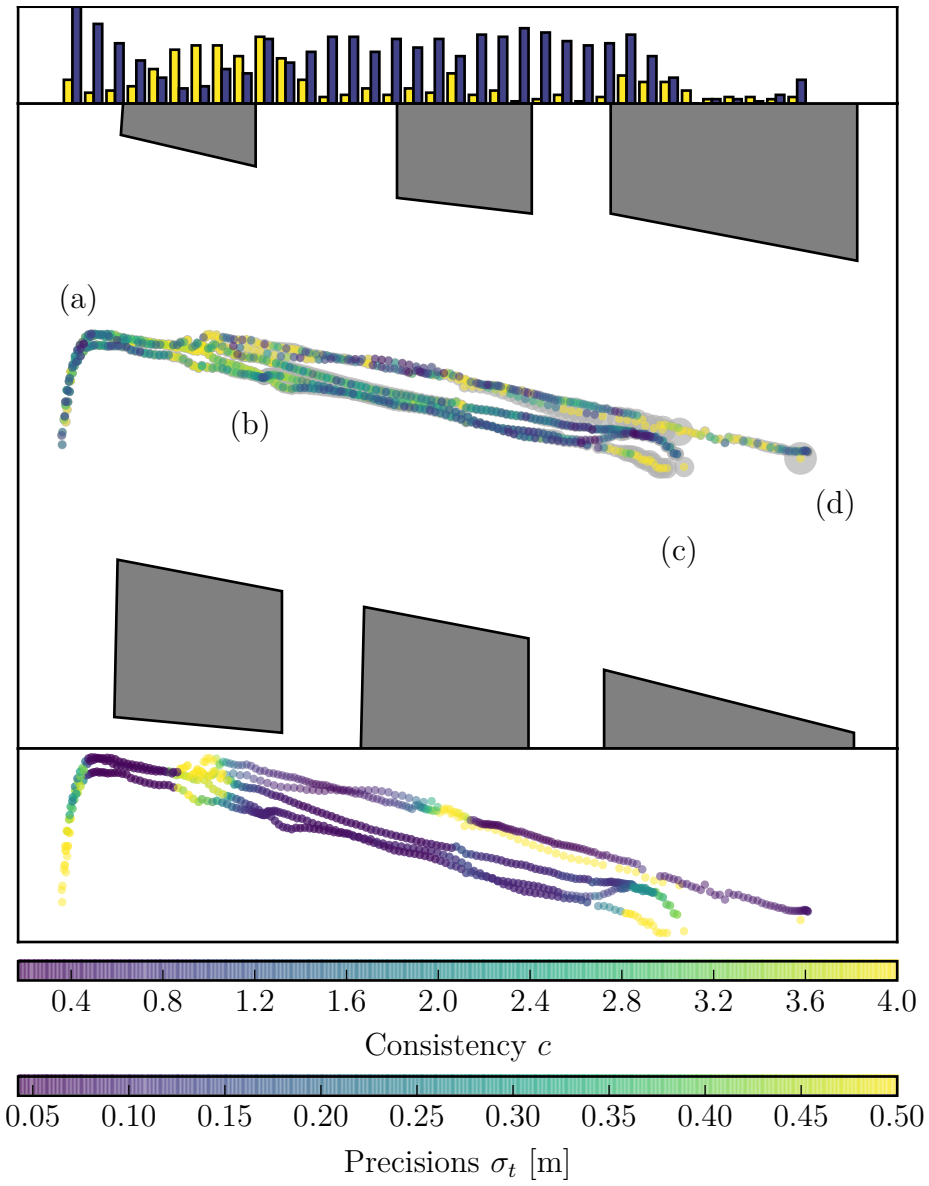


Figure 5.16: Bird's eye perspective of the annotation results within the scene. Each point corresponds to the (x, y) components of the groundtruth camera positions of it's respective target frame. Each point is bedded on a gray circle denoting the distance to it's reference frame. The spatial scales are not equidistant. The six boxes denote the houses from which we draw most of the annotatable tracks. The general area covered by the plot corresponds to 70 m in width and 50 m in height. Plots from top to bottom show the distribution of inliers and outliers, color coded consistency c and color coded precisions σ_t. Points of interest have been labeled in the image. This figure demonstrates the presence of local coherences on the annotation quality within sequences and the scene.

of outliers can be seen at position (b). Here, we not only see a high count of outliers but also a consistent decrease in the count of inliers. Both driving directions exhibit this behaviour. We first look at the driving direction from (a) to (d). The corresponding example frame in figure 5.17 shows that this corresponds to a position where the vehicle is in between one column of houses. Many of the available tracks start to disappear as the window corners walk out of view. The next left house is covered by a tree so the only region where new tracks become available are on the right side. The other driving direction runs out of possible tracks here as well, since this is the last part of the scene where reference tracks exist for the given driving direction. The low track count is confirmed by the track figure 5.18. We already identified objections to the annotation for low track counts in section 5.3 on page 54. These present observations confirm that this part of the scene is a point of failure due to sparse reference and target tracks.

Considering the initial guess of the frame matching, we see a small distance to the reference frame. This shows that the number of outliers can be attributed to the feature matching step. We can understand this by looking on the constellations of feature points. In both cases, we can only rely on tracks located on house facades parallel to the street. These points are arranged highly repetitively on a grid and are highly co-planar. Therefore, a translation parallel to the street leads to a very similar viable geometric constellation for tracks. Hence, the feature matcher can confuse the tracks and annotate a completely different set of tracks in the target frame. This systematic shift cannot be detected by RANSAC either, because the pose estimation would yield precise results given the completely wrong annotations. This can also be observed in the precisions presented in figure 5.16 where we observe very low precisions in the area around (b). We see that we can identify a systematic mismatching due to highly repetitive patterns in this region of the scene.

In contrast to the co-planar patterns, we see another constellation of tracks in (a). Here, we are looking on two house facades that are perpendicular to each other. Considering the low track count, we still see good annotation results at this point. The small consistency values might be a consequence of the high precision values. However, looking at the low error rate shows that the annotations are predominantly correct. Hence, the precision is the best we can get regarding the low track count. From these observations we can learn two things:

First, in this context, the absolute pose distance assumes higher values. Nonetheless, good annotation results in terms of error rates have been achieved at this point. Our quality measure *consistency* shows consistent results here as well. Hence, the examples (b) and (a) confirm the need for a precision-scale invariant quality measure. Second, the constellation of feature tracks has a big impact on the overall performance. In the curve, we see non-co-planar patterns. These give rise to less ambiguity in the possible point constellations. We learn that the choice of tracks should be as diverse as possible, such that many non-co-planar objects are to be annotated. This provides a better automatic annotation result but also improves the quality of the pose estimation in general.

The second largest mode in the outlier distribution can be seen in (c). We

observe several camera paths that show consistently false annotations or consistently good annotations, but nothing in between. We also observe higher reference frame distances, denoted by the larger gray circles. This suggests that the reason for the aggregation of outliers is also subject to sequence content. The lower camera path consisting of consistent outliers origins from the annotation of the forward-directed 0_0055 sequence. This sequence reaches further into the end of the street than other sequences in the same direction. In this case, the reference frame that is used for annotation falls far behind the actual camera position, which can also be deduced from the larger radius of the circles. From this case, we learn that the target sequence must be fully contained in the reference sequence for good annotation results.

The sequences starting in (d) also contribute to the mode of outliers. We see two paths that can be better distinguished in the other scatter plots. The complete path we see belongs to 0_0013. It shows periodically good consistencies, abruptly changing to bad consistencies and abruptly rising again. The target sequence in these cases was recorded with wipers turned on. Wipers make all existing tracks disappear. Hence, after each wiping event, a completely new set of tracks needs to be found, thereby breaking the temporal connection between the frames. The postprocessing makes use of majority votings for assignments. The disappearance of all tracks at once highly affects the postprocessing step and causes the observed high consistency value peaks.

The other path is sequence 0_0000 that has been annotated using the wiper sequence. We see that between (d) and (c) no annotations could be established. Looking at the gray circles, we see an increase in the radius for points that were annotated. This denotes higher reference camera distance and suggests to take a look at the similarity map which is shown in figure 5.17 on the facing page. From the similarity map we deduce that the path through the map has been found properly. However, the path is far away from being diagonal. This denotes a large speed difference between both sequences. The speed difference can be best seen in the error rate plot, where the points for the lower path are far more distantly placed from each other. In this case, the frame subsampling has not been chosen well. The camera distances between two of the subsampled frames are too large for the feature matching to work.

One more thing that needs to be discussed is the annotation error rate that is consistently high within the 75% percentile of 24% for consistency-inliers. In section section 5.3 on page 54, we learned that feature matching only provides useable results when the error rate is below 20%, which is slightly violated here. This shows that during the outlier detection step we also capture a reasonable amount of false annotations. Nonetheless, we see good annotation qualities all the way. To resolve this contradiction, it is must be noted that the camera poses are estimated using RANSAC, which is able to identify these outliers.

We can understand the error rate by revisiting the step. For each target track t_m, we collect all possible annotations that were proposed using the frame wise step. We then choose the landmark that has a simple majority. Hence, once a track has been assigned incorrectly, this annotation will remain in the set of candidates. This shows

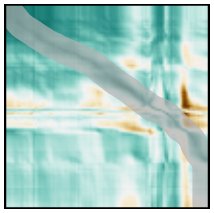
(a) Curve

(d) Wiper

(b) In between houses

(b) Backward end

(c) Forward end



Simiarity Map

Figure 5.17: Illustrative examples of frames which have a significant effect on the annotation results.

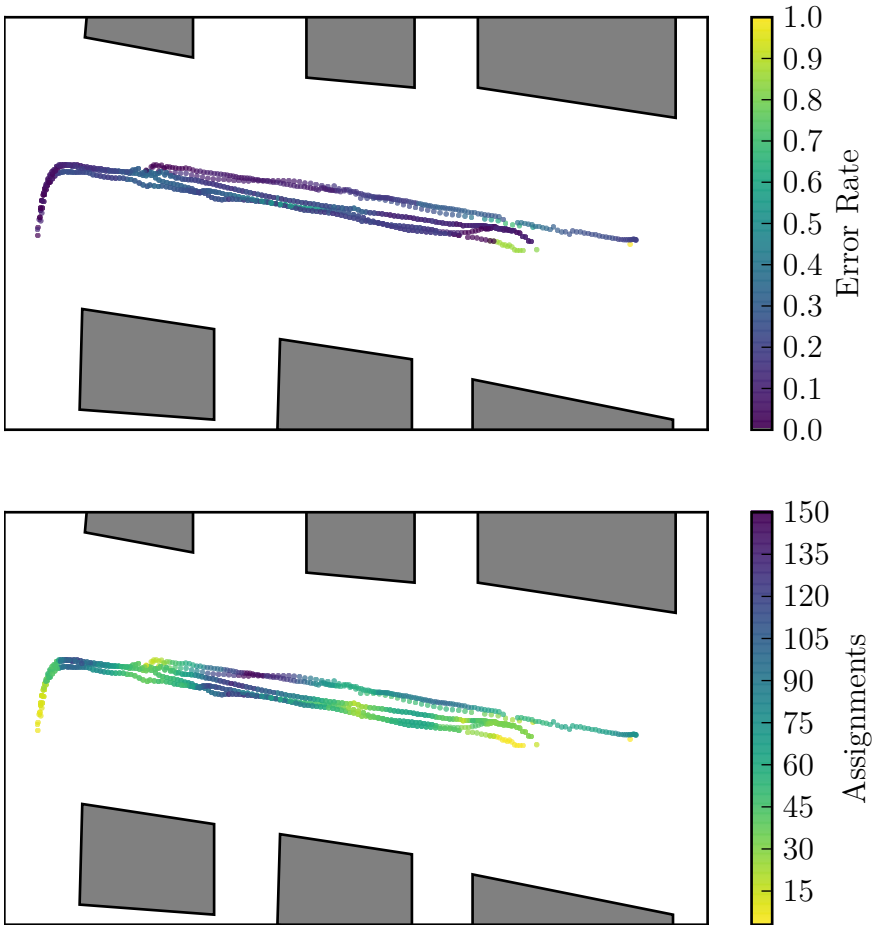


Figure 5.18: Extension to figure 5.16 on page 66. The plots from top to bottom show error rate and number of assignments.

that postprocessing is truly a crucial step and that it performs well by averaging over time. We conclude that the requirement on locality of the feature matching is a strong assumption and needs to be challenged again.

In summary, it could be shown that most of the inconsistent poses of intra-season annotations found in figure 5.5 on page 47 are due to difficult parts in the scene.

5.7 Time Consumption

In the following, we take up the time saving discussion from the introduction of the chapter again. We aim to assess the total required work that still needs to be done when using the presented method. The estimates are obtained by extrapolating from the test-set to the complete dataset and as such cannot give a precise quantification but only a coarse overview.

From section section 5.1 on page 44, we learned that the proposed annotation method produces approximately 68% useable annotations in the set of test sequences. In the subsequent sections, we examined reasons for the outliers and identified two main causes for inconsistent auto-annotations, namely poor frame matching and generic scene-related as well as sequence-related difficulties.

In the context of frame matching, we could identify inter-season sequence pairs as the root cause for poor alignments. In our case, this means that we have to provide a reference sequence for different season and lighting conditions. We have 4 recording days with 2 directions per sequence. Considering two light settings, namely day and night makes another 2 sequences per day. This totals to 16 sequences that are required to be annotated by hand in order to provide enough reference data.

From 200 total sequences we therefore are left with 184 sequences that need to be auto-annotated.

Using the outlier rate of 32 % and the required time of 7 h per sequence we can estimate the required time to 16·7 h for the reference sequences plus the remaining outlier frames with 410 h giving a total of 410 hours.

So far, we assumed that each consistency-outlier frame needs to be completely reannotated. From the former discussion, we also learned about the annotation error rate. This error rate is determined with respect to the number of found correspondences per frame-pair. Depending on the error rate, not all the 2D-to-3D correspondences need to be reannotated. We analyze how many of the correspondences are wrong in order to give a better estimate for the remaining manual interaction. We can condition the results on consistency-outliers again and find that the error rate is 32% within the 75 % percentile. This error rate gives an upper estimate for 75 % of the outliers whereas for the last 25 % the upper estimation can be attributed to a complete re-annotation.

We can combine these terms again. The time required to annotate the reference sequences stays untouched with 16·7 h whereas we find that the remaining annotation work reduces to 203 h, giving a total time consumption of 315 h.

In comparison to the required time for complete manual annotation of 1400 hours

we save approximately 1084 hours or 135 working days and reduce the workload to 22 % of the initial required workload for complete manual annotation.

6 Conclusion & Future Work

In this last chapter we want to conclude our examinations of the developed method and discuss how these results can affect future work. To this end, we first summarize the obtained results.

We presented a method that can be used to bootstrap the problem of aligning 2D video sequences to 3D geometry that was successfully applied to generate the reference dataset presented in [24]. First, coarse frame alignment of new video sequences with an annotated sequence is done to estimate an initial pose of the camera with respect to the point cloud. On average this pose has an accuracy of 3 m within the 75 % percentile. Using this pose as an initial guess control point to feature track annotations are obtained by solving a graph matching problem. Results indicate that this pose is consistent with the ones obtained by manual intervention in 30 % of cases. After the post processing step that removes outliers, we see that the performance is on par with manual annotation, resulting in a 5 −fold speedup from 420 min to 90 min minutes per sequence.

For the method to work, we do, however, require a complete manual annotation for different times of year or lighting conditions. Concluding, we presented a method that combines the best of two worlds: Quality assurance of manual annotations and the speed of an automated technique.

The discussion about future work will be split up into specific improvements to this method. The second part addresses quality assurance for the large-scale generated groundtruth.

From the examinations we also learned about objections to the method. One was found in the frame matching step. Here, in extreme cases we found that the frame stepping was not sufficiently well chosen. From our preliminary experiments we found that a constant mean velocity corresponding to the diagonal of the similarity matrix has been sufficient in most of the times. Further experiments should pick up the local estimation of the mean velocity as mentioned in section 4.3 on page 33. This gives rise to a better estimation of the true path, even in a context where the speed of the sequence is far removed from matching the diagonal path.

Another deficiency has been found in the combination of feature matching and postprocessing. In our requirement engineering we concluded that we want to avoid global regularization. However, experiments strongly indicated that regularization over time is a critical factor. This can be seen in two facts. First, we identified consistently high error rates in the annotations that contradicts the requirements found in section 5.4 on page 59. Here, we concluded that postprocessing brings noise into the annotations due to the unconstrained majority voting mechanism. Second, we saw problems in images with high repetitive structures. A long-ranging

regularization in this case can help as the ambiguities would be better resolved. Hence, further work can be put into a graph matching that takes several frame pairs into account. A more optimal solution would certainly be found by optimizing over a whole sequence. However, this would once more contradict our requirements on local manual intervention. Hence, finding a trade-off between regularization over a frame-pair and a whole sequence should be considered.

The authors of [15] only evaluated their groundtruth pipeline on a very small subset of all sequences, due to the time-scaling constraint that has been solved by the method developed in this thesis. Therefore, we now need to compare the batch generated groundtruth for useability.

So far we only compared camera poses for consistency, since this was the most groundtruth-related result that could be used on a larger scale. In this context, we compared our method against manual annotations. However, the question is whether the manual annotations are already precise enough to generate high accuracy groundtruth. Due to the lack of available data this question has not been addressed so far.

Hence, the next step would be to generate groundtruth for all registered sequences and compare the results. To this end, simple visual inspection of depth-to-visual edges can be used. More quantifying approaches could be warping images using the generated groundtruth and comparing images then in the visual domain. Here, we can once more identify difficulties within the method in order to improve the reference annotations. By incorporating these improvements we finally can ensure high quality groundtruth generation on a semi-automated basis.

Bibliography

[1] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. http://ceres-solver.org.

[2] Herbert Bay et al. "Speeded-Up Robust Features (SURF)". In: *Comput. Vis. Image Underst.* 110.3 (June 2008), pp. 346–359. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2007.09.014. URL: http://dx.doi.org/10.1016/j.cviu.2007.09.014.

[3] Daniel J Butler et al. "A naturalistic open source movie for optical flow evaluation". In: *Computer Vision–ECCV 2012*. Springer, 2012, pp. 611–625.

[4] Timo Dickscheid, Thomas Läbe, and Wolfgang Förstner. "Benchmarking automatic bundle adjustment results". In: *XXI. ISPRS congress, Beijing, submitted, accepted*. Citeseer. 2008.

[5] Ferran Diego et al. "Video alignment for change detection". In: *Image Processing, IEEE Transactions on* 20.7 (2011), pp. 1858–1869.

[6] Georgios D Evangelidis and Christian Bauckhage. "Efficient subframe video alignment using short descriptors". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35.10 (2013), pp. 2371–2386.

[7] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". In: *Commun. ACM* 24.6 (June 1981), pp. 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692. URL: http://doi.acm.org/10.1145/358669.358692.

[8] Xiao-Shan Gao et al. "Complete Solution Classification for the Perspective-Three-Point Problem". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 25.8 (Aug. 2003), pp. 930–943. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2003.1217599. URL: http://dx.doi.org/10.1109/TPAMI.2003.1217599.

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, pp. 3354–3361.

[10] The HDF Group. *Hierarchical Data Format, version 5*. http://www.hdfgroup.org/HDF5.

[11] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Choix de documents en appendice. Cambridge: Cambridge University Press, 2003. ISBN: 0521540518. URL: http://opac.inria.fr/record=b1100613.

[12] J. Heikkila and O. Silven. "A four-step camera calibration procedure with implicit image correction". In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. June 1997, pp. 1106–1112. DOI: 10.1109/CVPR.1997.609468.

[13] Ryan S Kaminsky et al. "Alignment of 3D point clouds to overhead images". In: *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*. IEEE. 2009, pp. 63–70.

[14] Daniel Kondermann et al. "On Performance Analysis of Optical Flow Algorithms". In: *Lecture Notes in Computer Science*. Springer Science and Business Media, 2012, pp. 329–355. DOI: 10.1007/978-3-642-34091-8_15. URL: http://dx.doi.org/10.1007/978-3-642-34091-8%5C_15.

[15] Daniel Kondermann et al. "Stereo Ground Truth with Error Bars". In: *Computer Vision–ACCV 2014*. Springer, 2015, pp. 595–610.

[16] Karsten Krispin, Rahul Nahir, et al. *Heidelberg Open prOgramming Kit for Enhanced Reference Sequences*. Internal HCI Framework.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[18] John P Lewis. "Fast template matching". In: *Vision interface*. Vol. 95. 120123. 1995, pp. 15–19.

[19] Lingyun Liu and Ioannis Stamos. "Automatic 3D to 2D registration for the photorealistic rendering of urban scenes". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 2. IEEE. 2005, pp. 137–143.

[20] D. G. Lowe. "Object recognition from local scale-invariant features". In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 2. 1999, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

[21] Andrew Mastin, Jeremy Kepner, and Jonathan Fisher. "Automatic registration of LIDAR and optical images of urban scenes". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 2639–2646.

[22] Rakesh Mishra and Yun Zhang. "A review of optical imagery and airborne lidar data registration methods". In: *The Open Remote Sensing Journal* 5.1 (2012), pp. 54–63.

[23] Ee Sin Ng and Nick G. Kingsbury. "Matching of interest point groups with pairwise spatial constraints". In: *Proceedings of the International Conference on Image Processing, ICIP 2010, September 26-29, Hong Kong, China*. 2010, pp. 2693–2696. DOI: 10.1109/ICIP.2010.5651903. URL: http://dx.doi.org/10.1109/ICIP.2010.5651903.

[24] A Nony, M Ous, and A Uthor. "Anonymous CVPR submission (cf. supplemental Material)". In:

[25] Daniel Scharstein et al. "High-resolution stereo datasets with subpixel-accurate ground truth". In: *Pattern Recognition*. Springer, 2014, pp. 31–42.

[26] J. Shi and C. Tomasi. "Good features to track". In: *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*. June 1994, pp. 593–600. DOI: 10.1109/CVPR.1994.323794.

[27] Ioannis Stamos. "Automated registration of 3D-range with 2D-color images: an overview". In: *Information Sciences and Systems (CISS), 2010 44th Annual Conference on*. IEEE. 2010, pp. 1–6.

[28] Ioannis Stamos et al. "Integrating automated range registration with multiview geometry for the photorealistic modeling of large-scale scenes". In: *International Journal of Computer Vision* 78.2-3 (2008), pp. 237–260.

[29] James Taylor et al. "The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, pp. 103–110.

[30] Oliver Wang et al. "Videosnapping: interactive synchronization of multiple videos". In: *ACM Transactions on Graphics (TOG)* 33.4 (2014), p. 77.

[31] Wenyi Zhao, David Nister, and Steve Hsu. "Alignment of continuous video onto 3D point clouds". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27.8 (2005), pp. 1305–1318.

Erklärung

Ich versichere, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den .....