Cross-View Camera Pose Estimation

By Matching Local Features in 3D

Sviatoslav Voloshyn

Master's Thesis

TU Delft Delft University of Technology

Department of Cognitive Robotics

Cross-View Camera Pose Estimation
By Matching Local Features in 3D

Master's Thesis

Sviatoslav Voloshyn

August 20, 2023

Student Number:	4774361	
Project Duration:	November, 2022 - August, 2023	
Comittee Members:	Dr. J.F.P. Kooij,	TUDelft, Supervisor and Committee Chair
	Ir. Z. Xia,	TUDelft, Daily Supervisor and Committee Member
	Prof. Dr. D.M. Gavrila	TUDelft, Cognitive Robotics, Committee Member

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Faculty of Mechanical, Maritime and Materials Engineering (3mE)
Delft University of Technology

Cognitive
Robotics

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The concept of autonomous vehicles has a long and fascinating history that stretches back further than the recent publicity would suggest. NavLab 5, the first truly self-driving vehicle was developed through a series of projects called VaMoRs back in 1980s [1]. Already in 1995 the vehicle, which supported lane keeping, was driven 98 % of the time autonomously over more than 600 miles from Pittsburgh to Los Angeles [1].

Over the last three decades, along with the development of computing capabilities, research in the area of autonomous vehicles has seen rapid growth. Thus, one generation of automation experts after another used to predict that truly self-driving vehicles will become widespread within the next couple of years. However, the reality is that the development of autonomous vehicles is still in its infancy. The current state of the art is that autonomous vehicles are only able to drive themselves on a limited number of roads and in a limited number of conditions.

Widespread deployment of autonomous vehicles in the real world, nevertheless, has clear-cut benefits for society as a whole. A large comprehensive review of state-of-the-art results for autonomous car technology [2] concluded that "the major benefits of autonomous cars include, but are not limited to, improving safety for both passengers and outsiders (pedestrians and other vehicles), new business opportunities, ease of use and convenience for people who cannot or do not want to drive, improved traffic conditions, and creating a consumer-centric experience".

The main obstacle to the widespread deployment of autonomous vehicles is the unpredictable and complex environments that they have to operate in. Thus, the main challenge is to develop a system that can handle both the uncertainty of the environment and the vehicle's sensors.

One aspect of such uncertainty lies in absolute vehicle pose estimation, which refers to the process of determining the position and orientation in in the global coordinate system, such as a global map of the environment [3].

1-1 Background and Motivation

Nowadays, the vast majority of vehicles use the Global Navigation Satellite System (GNSS) for absolute localization, since it provides a global location estimate, which does not deteriorate with time [4].

Table 1-1: GNSS Error Sources [5]

Effect	Error values
Ionospheric effects	± 5 m
Shifts in the satellite orbits	± 2.5 m
Clock errors of the satellites' clocks	± 2 m
Multipath effect	± 1 m
Tropospheric effects	± 0.5 m
Calculation and rounding errors	± 1 m

1-1-1 Hindrances of GNSS-based Localization

However, due to a number of factors, presented on Table 1-1, the GNSS error can be inaccurate. Particularly detrimental for autonomous driving is the multipath effect, the influence of which is underestimated in Table 1-1, as it is not specifically representative for vehicle localization. When a vehicle drives in an urban canyon, the GNSS signal traveling from the satellite can get reflected off the buildings or tall trees and reach the receiver with a delay. This delay causes the receiver to calculate wrong distance to the satellite, thus resulting in a large localization error, up to 10 m due to the multipath effect alone [6, 7, 8].

In an effort to alleviate the aforementioned hindrances of using GNSS for localization a few approaches exist. Some fuse GNSS data using temporal filters, such as a Kalman [9] or a particle filter [10]. Some [11, 12] further improve the fusion by adding Inertial Navigation System (INS) data . Other approaches [13, 14]combine multiple GNSS receivers and fuse their results to filter out outliers. In mobile robotics applications, geo-referenced fiducial markers can also be used to improve the localization error [15].

Nevertheless, these solutions are either too limiting in terms of their requirements, thus making them impractical for real-world applications, or they are still not able to achieve sub-meter accuracy with few outliers, required for autonomous vehicles.

In fact, the earliest works [16] on improving the GNSS errors date back to the early 2000s. Over the past two decades, researchers have been unable to significantly address the GNSS limitations, which is why alternative solutions became more prominent.

1-1-2 Current State-of-the-Art for Autonomous Vehicle Localization: HD Maps

One alternative to GNSS localization is use of HD maps [17]. These are highly accurate, 3-dimensional representations of the environment with semantic labels, which contain dense point clouds [18, 19] and/or LiDAR intensity images.

Both types of HD maps can be used either with a LiDAR point cloud or with a camera image to localize a vehicle. In the former case, when the HD maps contains only the LiDAR point cloud, localization reduces to 3D point matching, which is typically addressed through the Iterative Closest Point (ICP) algorithm [19] or its variants, such as the Generalized Iterative Closest Point (GICP) [18]. When the HD maps are constructed from intensity images, metric learning approaches with two Siamese branches embedding the LiDAR sweep and the intensity map, dominate the field [20, 21, 22, 23].

In efforts to eliminate the stringent requirement to have an expensive LiDAR sensor attached to the vehicle, some works focused on localizing with respect to an HD map using a camera image instead of a LiDAR point cloud. These typically also employ structure-based methods, which "represent the scene by a 3D model and estimate the pose of a query image by directly matching 2D features to 3D points"[24]. Early works in this area [24, 25, 26] extracted image features in a dense, pixel-wise manner. Recently, the state-of-the-art [27, 28, 29] has geared towards learning local, sparse descriptors due to the high computational complexity of finding pixel-level correspondences.

HD map-based localization methods are currently state-of-the-art for vehicle localization. The median error for the vehicle's location in all of the aforementioned methods is lower than 0.5 m on various datasets, which is a large improvement over the GNSS-based localization.

1-1-3 Limitations of HD Maps for Vehicle Localization

However, they also suffer from a limitation in terms of their applicability, as they require HD maps. First of all, HD maps take up a lot of space. Storing a LiDAR intensity map as a 16-bit PNG file would require approximately 900 GB for the city of Los Angeles [30]. Storing such amounts of information on board of a vehicle is infeasible for more than a single city.

This problem was addressed by [30] and [17]. The former introduced a learnable compression algorithm, which extracts features from the input HD map using a fully-convolutional network, binarizes them and stores them as a binary map. However, even despite a ∼400 times compression rate, the compressed maps are still non-negligible, in addition to the worsened performance. Furthermore, such compression is only applicable for localization through LiDAR sweeps and is unlikely to be feasible for camera-based localization. On the other hand, [17] suggest that it could be possible to download relevant map segments on the fly, relaxing the requirements for onboard storage. However, [17] evaluate that it would only be possible with a 5G connection, due to the large bandwidth requirements.

There are several additional limitations of HD maps. They are expensive to create and maintain. Some companies charge as much as $5000 per 1 km of mapping services in the US [31]. Furthermore, the HD maps need to be updated frequently not only due to the changes in the environment, but also to exploit the new generation of LiDAR sensors [20]. Scalability is also limited by the intensity calibration required to mitigate the differences in intensities across different beams and manufacturers [20]. Environmental factors, such as temperature, further affect the measured intensity [20].

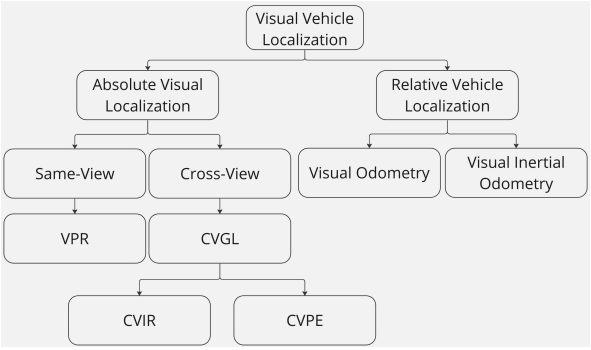


Figure 1-1: Overview of the Visual Vehicle Localization field

1-1-4 Visual Localization as an Alternative to HD Map-based Localization

Given the aforementioned limitations of HD maps, the benefits of a localization method that does not require an HD map are apparent. In this work we will focus on visual localization and more specifically on Cross-View Geo-Localization (CVGL) methods, which use satellite imagery as a reference.

Definitions

Visual localization refers to a family of localization methods, which determine the camera pose using the input of the on-board camera alone. The reduced overview of the field is shown on Figure 1-1. In absolute visual localization the camera pose of the query image is determined relative to a global visual representation of a known scene, such as a database of geo-referenced images. Absolute visual localization can be done either in the same view or cross-view. When both query and reference images are extracted in the same, ground view, it is referred to as Visual Place Recognition (VPR) [32]. When aerial imagery is used as a reference, it is referred to as Street-to-Satellite or CVGL. One family of CVGL methods is called Cross-View Pose Estimation (CVPE), which is characterized by determining the camera location within a correct aerial patch, as shown on Figure 1-2.

Relative visual localization methods, on the other hand, output the location of the vehicle relative a non-global reference, such as an image associated with a different viewpoint. These include Structure from Motion (SfM) techniques such as Visual Odometry (VO) and Visual-Inertial Odometry (VIO) [36, 37]. While these techniques are also vision-based, they are not of interest to this thesis work, as they are only able to provide relative pose estimates.

Benefits of CVGL over VPR

In absolute visual localization, VPR deals with a conceptually simpler problem due to the small viewpoint differences compared to CVGL. Thus, VPR primarily learns to discount weather and lighting changes between matching images, while CVGL needs to additionally learn cross-view correspondences in spite of the 90° view-point changes. One large benefit of CVGL, however, is that dense aerial imagery is available across most of the globe and is being regularly updated, while ground-view imagery is only available in a limited number of

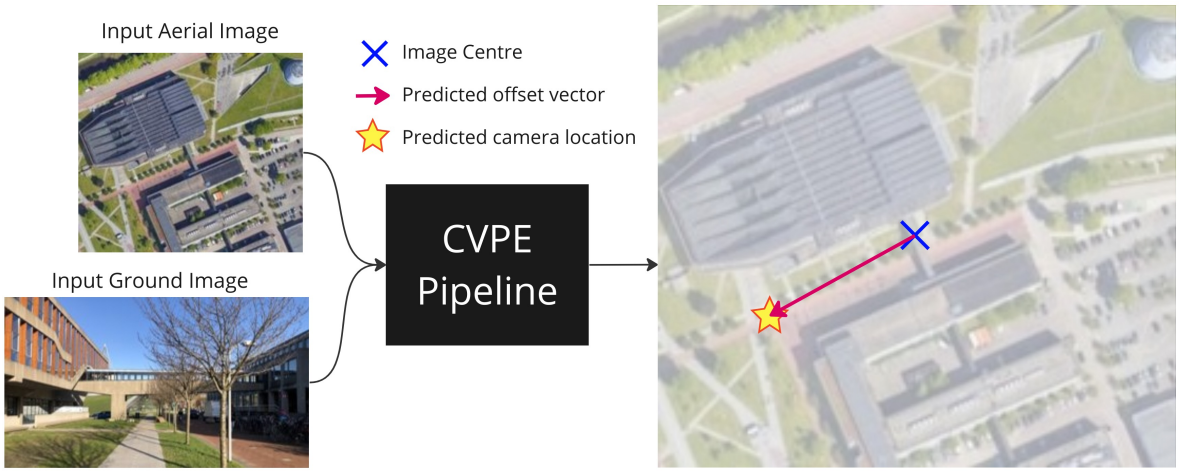


Figure 1-2: Overview of the CVPE pipeline. The camera pose is typically predicted as 2D offset from the aerial image centre. Most works, e.g. [33, 34, 35] also predict vehicle orientation.

locations and becomes quickly outdated. Therefore, CVGL presents itself as a more scalable solution for the problem of vehicle localization.

1-1-5 Problem Statement

The goal of this thesis is to explore whether sub-meter level accuracy in CVGL can be reached using camera input alone through, without the need for an HD map or other sensors, such as a LiDAR. Localization speed will also be a large factor for the candidate methods, as the localization should be able to run in real time on an autonomous vehicle. More specific constraints on the accuracy and the update rate requirements are also subjects to the related works section of this thesis.

1-2 Brief Overview of CVGL field

In this section we provide the brief overview of the CVGL field, which will be expanded upon in chapter 2. The CVGL can be divided into two distinct sub-fields: Cross-View Image Retrieval (CVIR) and CVPE. The former encapsulates the majority of the works in the field and is characterized by retrieving the correct satellite patch, which contains the query ground image. The latter, on the other hand, is characterized by retrieving the correct vehicle location within the satellite patch, which is usually already known. CVPE is a smaller sub-field, which only recently gained traction.

The CVGL works by large deal with creating image descriptors, which are similar for a positive match, despite the viewpoint differences. Bridging the aerial to ground domain gap is the largest challenge in the field and is usually addressed using either geometric transformations, such as the homography assumption, or learning-based approaches.

The CVPE field mostly uses local features to match parts of the ground image with parts of the aerial image to determine the vehicle location [38, 35, 34]. Apart from this, some approaches

also use global features, which are transformed across the domain gap using either geometric transformations [33] or learning-based approaches [39].

1-3 Limitations of Current CVGL Methods

We use the conclusions of the related works to guide the research directions. In short, the literature study concluded that:

- There is a mismatch between the existing localization setting and the one required by the industry. A real-life vehicle is able to ensure a decently accurate localization prior using GNSS for instance, so assuming a $20\,\text{m} \times 20\,\text{m}$ uncertainty region on the camera pose is unjustified. On the other hand, the required localization performance needs to be improved significantly to match the industry standard.
- When trained on publicly available datasets, the state-of-the-art CVGL methods demonstrate generally poor performance on the cross-area setting, some close to random. Moreover, it is likely that the road prior is the most important factor for any incremental improvements in performance.
- Most methods use local features. Local features generally result in better performance in terms of both accuracy and speed. Furthermore, local features are more interpretable than global features, which is important for understanding the model's behaviour. Nevertheless, when supervision occurs at the camera pose level, interpretability is limited, since the effect of local matches on the camera pose is latent.
- Existing works, most of which use local features with image-level supervision or global features entirely, are not data efficient, which is a limitation, given the relatively small sizes of existing, publicly available datasets.
- Methods that use iterative pose refinement instead of the hypothesis formulation perform worse. This is due to the fact that the iterative pose refinement is not guaranteed to converge to the global optimum as well as the slow inference speed.
- Longitudinal performance of the state-of-the-art is significantly worse than lateral. Due to the use datasets with only forward facing cameras in all but one methods, the longitudinal performance is largely dependant on the implicit depth prediction capacity of the backbone, which is typically poor with respect to the accuracy requirements.

These conclusions are further reiterated in section 2-5.

1-4 Research Questions

In an effort to address the problem in a structured way, the main research question (**MRQ**) is defined as follows:

> **"How can Cross-View Geo-Localization methods using camera input alone provide localization performance sufficient for autonomous driving?"**

Based on the conclusions of the literature study, summarized in section 1-3 and elaborated on in chapter 2, we define a set of sub-research questions to help shape the research directions:

SRQ1: "How can the localization setting be refined to tailor to the requirements of autonomous driving?", i.e. "What is the fine-grained localization setting?"

SRQ2: "How can local feature matching be optimized for the fine-grained localization setting?"

SRQ3: "How can interpretability of CVGL be improved?"

SRQ4: "How can data efficiency of CVGL be improved?"

1-5 Chapter Overview

In chapter 2 we review the related works in the field of CVGL. We start by defining the localization requirements, followed by the overview of literature in the CVIR and CVPE fields. We then discuss the datasets and metrics and conclude with the list of thesis' contributions.

In chapter 3 we start by defining the experimental setup and proceed with defining the baseline that will be used. Afterwards we describe the methodologies of the three methods proposed in this thesis along with implementation details. We conclude the chapter with an overview of hypotheses, which will be used to drive the experiments in chapter 4.

In chapter 4 we address the mentioned hypotheses through a series of experiments. We report both qualitative and quantitative results as well as unexpected insights. We conclude the thesis in chapter 5, where future works are also discussed.

Chapter 2

Related Works

The field of Cross-View Geo-Localization (CVGL) can generally be divided into two families of approaches: Cross-View Image Retrieval (CVIR) and Cross-View Pose Estimation (CVPE). CVIR is the older formulation of the two and is concerned with retrieving aerial matches to a ground image from a geo-referenced aerial image database. The overview of the architecture is shown on Figure 2-1. Most notable disadvantages of the CVIR approaches are: (1) the irreducible error due to the sampling density of aerial images, (2) the inefficiency associated with computing similarity between descriptors of many largely overlapping aerial patches.

In CVPE the pose of a query ground image is estimated with respect to the centre of a correct, geo-referenced aerial image match. This addresses both limitations of CVIR methods. The absence of the assumption that all content in the aerial image must match all content in the ground image allows for more interpretability and more native orientation estimation. The overview of the architecture was shown on Figure 1-2.

This chapter starts off with the overview of localization requirements in section 2-1. Afterwards, we briefly give an overview of the CVIR field in section 2-2 in order to draw general

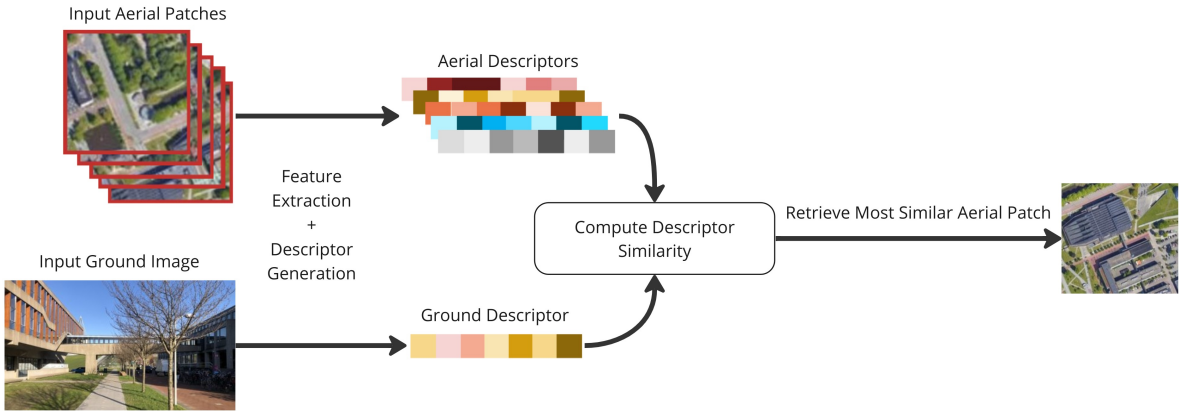


Figure 2-1: Generic CVIR architecture.

insights about the CVGL task. In section 2-3 we provide a more detailed overview of the CVPE field, since it is more relevant to our problem. More specifically, we compare a small number of state-of-the-art methods. Afterwards, the relevant datasets and metrics are introduced in section 2-4. Finally, we conclude the chapter with a summary of the literature study in section 2-5. Thesis' contributions are listed in section 2-6.

2-1 Localization Requirements

In order to answer the main research question **MRQ:** "How can Cross-View Geo-Localization methods using camera input alone provide localization performance sufficient for autonomous driving?", we need to define what "sufficient" means in the context of autonomous driving. To do so, in this section we provide the overview of localization requirements. First, the requirements for localization accuracy are discussed in subsection 2-1-1. Then, the requirements for localization speed are discussed in subsection 2-1-2. Finally, a summary of the requirements is provided in subsection 2-1-3.

2-1-1 Localization Accuracy Requirements

The localization accuracy requirements for an autonomous vehicle depend on the purpose for localization, which in turn depends on how autonomous a vehicle is. The most stringent requirements on absolute localization error are set by vehicle communications and the active control for autonomous vehicles beyond what Society of Automotive Engineers (SAE) [40] define as Level 3.

Vehicle-to-Everything (V2X) is a communications protocol designed for connected vehicles. Absolute localization requirements depend on the type of communications. For instance, [41] break down the requirements into which-road ($<$5 m), which-lane ($<$1.5 m), and where-in-lane ($<$1 m). In the United States the Federal Motor Vehicle Safety Standards in Vehicle-to-Vehicle (V2V) Communications report published by National Highway Traffic Safety Administration (NHTSA) determined that 2D position of the vehicle must be reported to an accuracy of 1.5 m (1σ), which corresponds to half of the width of a lane. This tentative accuracy constraint is motivated in [42] by the fact that "if vehicles provide position data within this level of accuracy, safety applications should be able to determine whether another vehicle is within its lane of travel".

The second source of localization accuracy requirements stems from the active control. Active control refers to the ability of a vehicle with an automation level SAE3 or higher to control the vehicle in the absence of the driver. Control commands result from a planning algorithm, which takes into account the location of a vehicle at a given time.

Reid et al. (2019) [43] take as reference a one hundred times improvement in road safety through the virtual driver system compared to a human driver. The resulting target fatalities per mile (2.5×10^{-10}) is similar to that of commercial aviation today. In order to calculate the required localization accuracy for such a target, [43] first propagate the influence of localization error on the control of a vehicle and then relate control block to the target level of safety, see Figure 2-2.

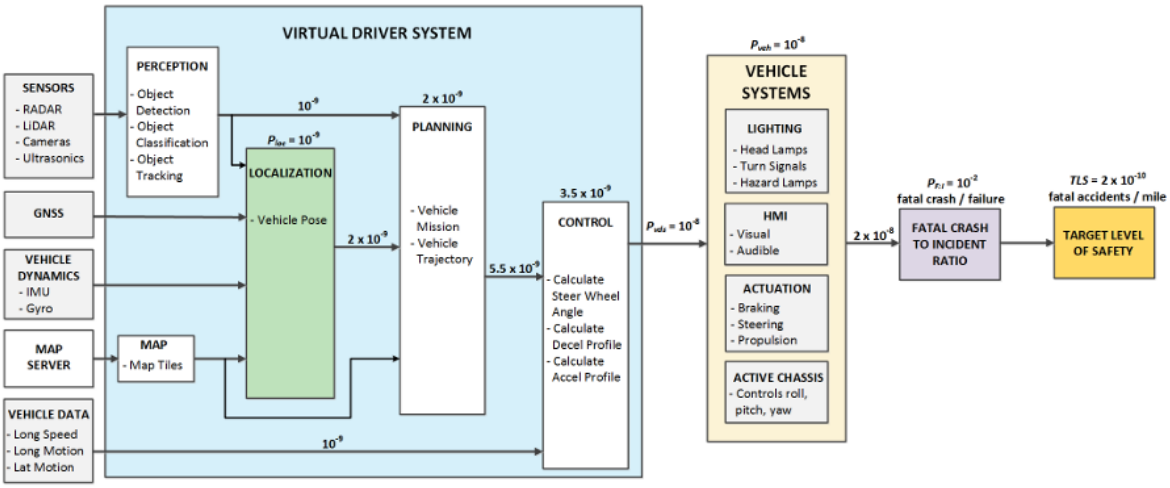


Figure 2-2: Virtual driver system integrity risk allocation from [43]

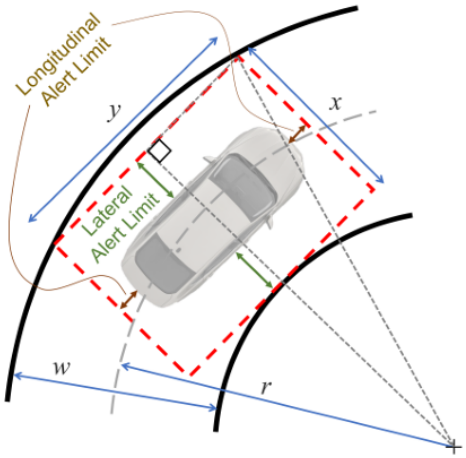


Figure 2-3: Bounding box geometry in a turn. This shows the allowable maximum position error of the vehicle to ensure it is within the lane known as the alert limits [43]

They utilize information regarding the road information (dimensions, curvature, speed limits) as well as the vehicle type for different road types and vehicles to construct the vehicle's bounding box, which includes the alert limits for when the vehicle leaves the lane, see Figure 2-3.

Using the required alert limits for different vehicles on different road types, calculated using Figure 2-3 and how likely it is that the vehicle will leave the lane and how likely such an event will result in a collision, the authors calculate the required localization accuracy to ensure a 100 times improvement in road safety. The results are shown in Table 2-1 and Table 2-2.

2-1-2 Localization Speed Requirements

The frequency at which the vehicle is able to globally localize itself is also important and may guide the design choices throughout this thesis work. A lag in position update leads directly

Table 2-1: Localization requirements for US freeway operations with interchanges. Assumes minimum lane widths of 3.6 meters and allowable speeds up to $137\,\mathrm{km\,h^{-1}}$. From [43]

Vehicle Type	Accuracy (95%)				Alert Limit			
	Lateral [m]	Long. [m]	Vertical [m]	Attitude [deg]	Lateral [m]	Long. [m]	Vertical [m]	Attitude [deg]
Mid-Size	0.24	0.48	0.44	0.51	0.72	1.40	1.30	1.50
Full-Size	0.23	0.48	0.44	0.51	0.66	1.40	1.30	1.50
Standard Pickup	0.21	0.48	0.44	0.51	0.62	1.40	1.30	1.50
Passenger Vehicle Limits	0.20	0.48	0.44	0.51	0.57	1.40	1.30	1.50
6-Wheel Pickup	0.14	0.48	0.44	0.51	0.72	1.40	1.30	1.50

Table 2-2: Localization requirements for US local roads. This assumes lanes $3.0\,\mathrm{m}$ wide with a minimum curvature of $20\,\mathrm{m}$ or $3.3\,\mathrm{m}$ wide with minimum curvature of $10\,\mathrm{m}$. From [43]

Vehicle Type	Accuracy (95%)				Alert Limit			
	Lateral [m]	Long. [m]	Vertical [m]	Attitude [deg]	Lateral [m]	Long. [m]	Vertical [m]	Attitude [deg]
Mid-Size	0.15	0.15	0.48	0.17	0.44	0.44	1.40	0.50
Full-Size	0.13	0.13	0.48	0.17	0.38	0.38	1.40	0.50
Standard Pickup	0.12	0.12	0.48	0.17	0.34	0.34	1.40	0.50
Passenger Vehicle Limits	0.10	0.10	0.48	0.17	0.29	0.29	1.40	0.50

to increased uncertainty in localization, predominantly in the longitudinal direction. For instance, if a vehicle traveling at $100\,\mathrm{km\,h^{-1}}$ localizes at a frequency of 10 Hz, the successive localization updates will be 2.7 m apart, which is clearly unacceptable based on Table 2-1 and 2-2.

Reid et al. also address this issue in [43] and conclude that the localization frequency depends heavily on the speed of the vehicle. At top operational speed of $130\,\mathrm{km\,h^{-1}}$ the update rate must be at least 200 Hz in order to ensure that update rate induced uncertainties make up only a small percentage of the total location uncertainty. At $130\,\mathrm{km\,h^{-1}}$ a 150 Hz update rate is sufficient.

However, [43] admit that it is not strictly a requirement on absolute localization. Relative localization can be used in-between global localization updates to produce accurate localization estimate on a short time interval. If a perfect IMU data is available on short time intervals, the update rate is limited by the IMU rate. Therefore, the true localization latency requirement is dependent on the IMU quality and update rate, which varies between vehicles and road conditions. For the purpose of this thesis work, we will assume that at least a 5 Hz update rate is required, such that the IMU operating at 200 Hz will predict a relative location at most 40 times between successive absolute localization updates.

2-1-3 Summary of Localization Requirements

In the previous two sections, we have discussed the requirements for localization in terms of accuracy and speed. To conclude this chapter we provide a short summary.

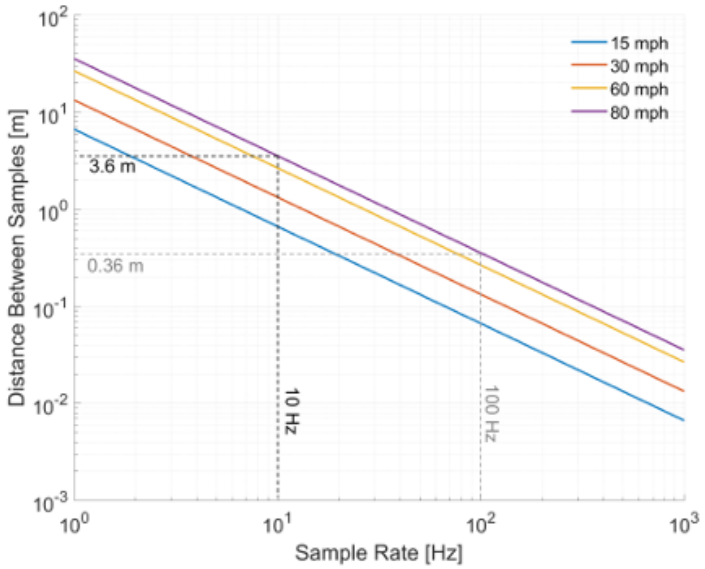


Figure 2-4: The relationship between sample rate, speed and distance between samples. From [43]

In terms of localization accuracy, it is clear that the requirements are dependent on the vehicle type and the road type. For the purpose of this literature study and the subsequent thesis, we will adopt the requirements for a "Mid-Size", defined by Table 2-1 and 2-2, as we expect the demand for automation of the most conventional vehicles to be the highest. Therefore, the localization accuracy, sufficient for intelligent driving is 0.15 m in each direction on a local road and 0.24 m and 0.48 m in lateral and longitudinal directions on a freeway, respectively. All values refer to the 95% confidence interval.

In terms of localization speed, it was concluded that it needs to be sufficiently high, though it was difficult to quantify. For the purpose of this work, we set the upper limit on the localization update rate as 5 Hz, so that relative localization is validated through absolute iterations with sufficient frequency.

2-2 CVIR Methods

Despite the fact that CVIR methods in computer vision, the retrieval formulation has a few inherent limitations, particularly detrimental to the fine-grained task. First of all, there is an irreducible error that has to do with the sampling density of aerial images. Furthermore, in the fine-grained setting the aerial images would look nearly identical, thus making it difficult to distinguish between them. Finally, the retrieval formulation can not produce a 6-DoF pose estimate.

Due to these reasons, we will not evaluate retrieval methods extensively. We will just outline the key ideas present throughout the literature that may guide the intuition for the subsequent CVPE methods.

Following [44], we systematically outline a baseline retrieval architecture. A generic image retrieval approach is demonstrated on Figure 2-5. As input it requires a query image and a

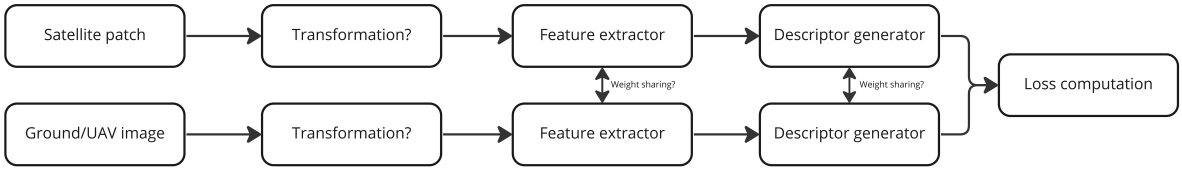


Figure 2-5: Generic CVIR architecture.

(a) Ground image **(b)** Polar transformed ground image

Figure 2-6: Polar transformation example

pool of aerial images. Depending on the dataset, the aerial images may be aligned or not aligned. Aligned imagery means that for every ground image there exists an aerial image in the pool which is centered at the query's location. In case of unaligned imagery, the aerial patches cover the location of the query, though it might not be centered. Furthermore, the aerial images can have various orientation offsets to simulate the vehicle's orientation ambiguity. The vast majority of research in CVIR is done on aligned imagery, which is sometimes randomly rotated within a small (up to 20°) range.

In the two sections below we highlight the choices that can be made for the transformation and feature extraction modules. The remaining modules are not relevant for CVPE and thus are only addressed in the preceding Literature Study.

2-2-1 Transformation Module

The transformation module on Figure 2-5 is designed to bridge the domain gap. Most works use some kind of transformation module to improve performance, which indicates the difficulty of the Cross-View Matching (CVM) task.

The transformation module can either bring the ground image to the aerial domain or bring the aerial images to the ground domain. Aerial-to-ground transformations are the most common in literature. There are two types of this transformation: Polar [45, 7, 8, 46, 47, 48] and Generative Adversarial Network (GAN)-based [49, 50]. Polar transformation is a geometric transformation which preserves all information, while sacrificing geometric consistency, see Figure 2-6. Methods that use the polar transformation are the most numerous and some [48, 46] perform close to the state-of-the-art for image retrieval.

Ground-to-aerial transformations are less common, since ground images are typically less informative, so further reducing information by projecting to the aerial domain is not beneficial. However, there are also two types of transformations to do so: Inverse Polar [51] and GAN-based [52]. Inverse polar transformation is shown on Figure 2-7. Both these methods are significantly worse than the state-of-the-art.

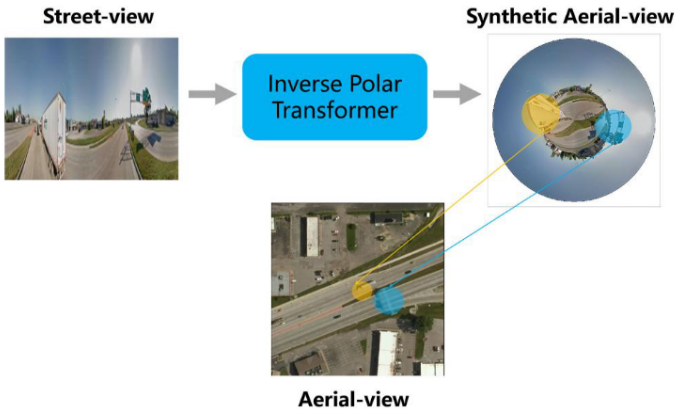


Figure 2-7: Inverse Polar Transformation example from [51]

Other works [53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63] did not use an explicit geometric transformation module. They did so for a number of reasons:

- The transformer backbone is used. A CNN is translation invariant, which means that two objects that are close on the original image will also be close on the extracted feature map. However, in the case of CVM this is not necessarily desired as objects close together on the ground image may be far apart on the aerial image. Transformers do not have this inductive bias and thus can bridge the domain gap implicitly. Examples include [57, 58].
- The domain gap is small, for instance when the ground image is taken with a Unmanned Aerial Vehicle (UAV), not a vehicle on the ground. Examples include [59, 60].
- Local features are used. Local feature matching can be explicitly engineered in a way that features from some part of the ground image match local features in the correct parts of the aerial image, taking geometry into account. Examples include [61, 62, 63].

2-2-2 Feature Extraction Module

Any existing pre-trained backbone can be used for feature extraction. Both CNN and transformer backbones were used. The CNN backbones are usually pre-trained on ImageNet.

Among the most popular CNN backbones are VGG-16 used by [45, 56, 64, 7, 46, 8, 65, 66, 63, 48] and ResNet used by [49, 60, 67]. The other option is PCPVT-5 used by [68]. Finally, some convolutional backbones are created and trained from scratch to taylor to the specific feature extraction needs, e.g. [54, 55]. Transformer backbones are used by [57, 58].

2-3 CVPE Methods

Despite being a younger field with only a handful of works, CVPE methods are more relevant to our problem, since they yield better localization results and are more suited to the fine-grained localization task.

The complete overview of the field is provided in the Literature Study. Furthermore, the in-depth methodology and analysis of each method is provided in Appendix A. In this section, a high level overview is provided with a goal to specifically address the strengths and limitations of existing literature.

Three methods can currently be considered the stat-of-the-art, depending on the evaluation metric and the dataset: UncertaintyAware [39], SliceMatch [34] and CCVPE [35]. Next to these methods we will also consider HighlyAccurate [33], which, despite not longer being the state-of-the-art, can be expected to excel in the fine-grained setting due to its iterative refinement.

Unlike in CVIR, it is quite difficult to define a generic architecture for CVPE due to the variety of approaches. However, in general, the methods can still be divided based on the transformation module used, the locality of the features and the pose estimation method, as described in subsection 2-3-2, subsection 2-3-3 and subsection 2-3-4 respectively.

Furthermore, we evaluate the impact of the quality of the ground truth in subsection 2-3-5 and the speed of the methods in subsection 2-3-6.

Prior to the above, however, we first present the quantitative performance of the select methods in subsection 2-3-1.

2-3-1 Quantitative Performance

Unfortunately, the four methods that are considered, i.e. UncertaintyAware [39], HighlyAccurate [33], SliceMatch [34] and CCVPE [35], were not evaluated on the same datasets.

However, the first two methods were compared on the FordAV [69] dataset by [39]. The results are shown in Table 2-4. Unfortunately, only retrieval errors are reported, which are not directly comparable to the more informative mean/median localization errors. However, it is clear that HighlyAccurate is significantly worse than UncertaintyAware. Interestingly, UncertaintyAware evaluated on the cross-area, cross-vehicle setting outperformed HighlyAccurate evaluated on the same-area, same-vehicle setting. Moreover, even without vehicle frames, i.e. when the ground image is "blacked-out", i.e. all values for the RGB channels are set to 0, the longitudinal performance of UncertaintyAware was found to be similar to HighlyAccurate with the ground image available. This indicates that the performance of the latter is not better than using only the road prior, i.e. predicting any camera pose location along the road.

The last three methods, i.e. HighlyAccurate [33], SliceMatch [34] and CCVPE [35], were compared on the KITTI [70] dataset with each work reporting their own results. The results are shown in Table 2-3. Clearly, both SliceMatch and CCVPE easily outperform HighlyAccurate on nearly all metrics. Between SliceMatch and CCVPE, the latter is significantly better in most metrics, especially in terms of the median error. The only exception is the orientation error on the cross-area setting, where SliceMatch is better. This is attributed to the fact that there is a larger focus on orientation performance compared to localization in SliceMatch. At test time SliceMatch uses 15×15 location hypotheses and 64 for orientation. During training, the values are lower, but the ratio is almost identical. On the other hand, CCVPE uses dense classification for position, i.e. for each of 512×512 pixels and 20 orientation bins. It is clear that SliceMatch targets orientation estimation and CCVPE prioritizes position.

Table 2-3: Quantitative comparison between SliceMatch [34], CCVPE [35] and HighlyAccurate [33] on the KITTI dataset. Results provided by each paper. Initial pose is chosen in $40\,\mathrm{m} \times 40\,\mathrm{m}$ area.

Same-Area		↓ Localization (m)		↓ Orientation (deg)	
		mean	median	mean	median
±10 deg	HighlyAccurate [33]	12.08	11.42	3.72	2.83
	SliceMatch [34]	7.96	4.39	4.12	3.65
	CCVPE [35]	**1.22**	**0.62**	**0.67**	**0.54**
No orientation	HighlyAccurate [33]	15.51	15.97	89.91	90.75
	SliceMatch [34]	9.39	5.41	**8.71**	**4.42**
	CCVPE [35]	**6.88**	**3.47**	15.01	6.12
Cross-Area		↓ Localization (m)		↓ Orientation (deg)	
		mean	median	mean	median
±10 deg	HighlyAccurate [33]	12.58	12.11	3.95	3.03
	SliceMatch [34]	13.50	9.77	4.20	6.61
	CCVPE [35]	**9.16**	**3.33**	**1.55**	**0.84**
No orientation	HighlyAccurate [33]	15.50	16.02	89.84	89.85
	SliceMatch [34]	14.85	11.85	**23.64**	**7.96**
	CCVPE [35]	**13.94**	**10.98**	77.84	63.84

Table 2-4: Quantitative comparison between HighlyAccurate [33] and UncertaintyAware [39]. Recall in percent on the first two scenes of the FordAV dataset is shown. Results provided by each paper. Initial pose is chosen in $40\,\mathrm{m} \times 40\,\mathrm{m}$ around the vehicle with up to $20°$ of rotation noise. Own work.

	Cross-area	Cross-vehicle	Log 1						Log 2					
			Lateral			Longitudinal			Lateral			Longitudinal		
			1.0m	3.0m	5.0m	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m
HighlyAccurate	✗	✗	46.1	70.4	72.9	5.3	16.4	26.9	31.2	66.5	78.8	4.8	15.3	25.8
UncertaintyAware	✗	✗	**96.3**	**99.6**	**99.6**	**76.0**	**95.3**	**96.0**	**88.0**	**99.9**	**100.0**	**58.9**	**93.3**	**93.6**
UncertaintyAware	✓	✓	77.0	96.2	97.6	24.0	67.6	76.1	73.0	96.5	97.8	25.6	61.7	69.4
UncertaintyAware w/o vehicle frames	✗	✗	15.1	1.3	72.0	5.0	15.2	24.4	11.3	37.8	62.2	4.7	15.3	26.0

2-3-2 Transformation Module

In section 2-2 three transformation modules were identified: transform aerial image to ground image, transform ground image to aerial image and transform both images to a common domain, also known as a hypersphere. In CVPE the four state-of-the-art methods also span all three methods.

Aerial to Ground (A2G) Transformation

The only method that uses A2G transformation is HighlyAccurate [33], which uses the projective transform described in subsection 2-2-1. As shown in Table 2-3, and especially in Table 2-4, HighlyAccurate's performance is significantly poorer than that of the state-of-the-art.

While in the CVIR setting the A2G methods were more dominant, this does not translate to CVPE. This is likely the case because in CVIR bridging the domain gap, i.e. transforming the features to the same positional layout was important to allow for geometrically consistent descriptors that could be compared directly for retrieval. In common CVPE formulation, the fact that the aerial image contains a match is a given. Thus, the need for a transformation is reduced, especially a static one that cannot predict depth and is information lossy.

Ground to Aerial (G2A) Transformation

The G2A transformation is used by UncertaintyAware [39], who claim to significantly outperform HighlyAccurate [33]. Unlike in CVIR, the projective transformation is not used. Instead, the authors create and iteratively refine the Bird's-Eye View (BEV) representation. This has three advantages over the projective transformation: (1) depth information is preserved, (2) the knowledge of camera intrinsics is not always required (though it is in the case of UncertaintyAware), thus the training is not constrained to only a few datasets, where the intrinsics are available, and (3) it is easy to scale the method to a camera rig as opposed to a single camera.

The authors of UncertaintyAware made use of recent advances in the fields of 3D object detection and scene segmentation to create a precise BEV representation. There are generally three ways to create a BEV representation using a ground image, i.e. perform a G2A transformation:

1. **Compression**, where the ground image features are first compressed along the height dimension and then expanded into the BEV. This method was typically used in earlier works in the field of 3D object detection, [71, 72]. The disadvantage is that the depth information is only implicitly modelled (by the depth embedding of the BEV features) and thus depth prediction is not as accurate.

2. **Lifting**, where the depth distribution per ground feature map pixel is predicted and features are pooled along pillars of infinite height to create a regular 2D BEV representation, similar to [73]. Some of the works that use this method are [74, 75, 76], but neither was done for the purpose of CVPE. The closest work to using lifting approach in CVPE

is [77], who used OpenStreetMap data instead of an aerial image and achieved superb performance. However, despite predicting depth explicitly, the feature aggregation to create a useful BEV representation seems to be a performance bottleneck.

3. **Query-based**, where instead of starting from the ground image and predicting 3D points, the 3D points of interest are used as query to look up information from the ground image. This is likely the largest family of methods of the three with [78, 79, 80] being the most widely known. In the context of CVPE, this method is used by UncertaintyAware [39]. The advantage of this method is that sequential attention blocks are able to refine the BEV representation iteratively, which is not possible with the other two methods, while the deformable variant of said attention is able to model small offsets without the need for a dense query grid. Despite the fact that UncertaintyAware uses deformable attention to refine the BEV feature map, many of the novelties from the field of object detection are not used, thus there is potential to achieve even better performance. One example of such a novelty is the Multi-Camera Deformable Cross-Attention module from [78], which has been shown to be superior when aggregating information from multiple cameras.

Pose Estimation on the Hypersphere

The remaining two methods, SliceMatch [34] and CCVPE [35] make use of geometric relationships between the ground and aerial images. SliceMatch makes use of the azimuth prior to match local features from ground image cells and aerial image rays. CCVPE uses a similar approach, but instead of matching slices on the aerial view, it matches descriptors at each grid location in the aerial image, which encompass all orientations, with the ground image. Circular descriptors are used for both the ground image and each hypothesis on the aerial image, which makes orientation estimation intuitive. Modelling geometric relationships is a promising direction, which clearly yields state-of-the-art results.

2-3-3 Type of Features

We identify two types of features: local and global. Local features are those that are dedicated to explaining a small region in the input image. Thus, aligning local features geometrically in a different way than how they were extracted from the input image is a powerful tool for bridging the domain gap. Global features are those that are extracted from the entire image and thus any geometric correspondences may be lost. In CVIR most methods used global features.

In CVPE, all methods that we review, except HighlyAccurate [33], use local features. The fact that HighlyAccurate demonstrates the worst performance is a strong motivation to work with local features.

The degree of locality is different per method, nevertheless. In UncertaintyAware [39] the aerial representation used for pose estimation is global, yet the construction of the BEV representation involves manipulating very fine-grained, pixel-level features. Per BEV query, the attention mechanism aggregates information from each ground image feature map pixel, thus the features can be seen as pixel-wise local.

In SliceMatch [34] both aerial and ground features are local, but not as fine-grained as pixel-wise. The aerial features are local per slice, while ground features are local per vertical cell. In CCVPE [35] the features can be seen as "least local" The aerial features are local per each of the 2D hypotheses, but global in terms of orientation. The ground features are global too.

Overall, there is not a clear-cut correlation between the locality of the features and performance. However, from the example of HighlyAccurate [33], it is clear that aligning global (despite projective transform) feature maps can not warrant sufficient performance.

2-3-4 Camera Pose Estimation Method

There are two ways to determine the camera pose: the hypothesis formulation and iterative pose refinement. The hypothesis formulation involves generating a number of hypotheses and evaluating how well does the aggregation of local features from the perspective of each hypothesis match the ground image. This is done by UncertaintyAware [39], SliceMatch [34] and CCVPE [35]. The iterative pose refinement involves two features maps, one describing a region in the aerial image $F_{A,i}$ and one describing a ground image F_G. If the aerial feature map is continuous and differentiable, then comparing $F_{A,i}$ with F_G not only yields a similarity score, but also a gradient, which can be used to update the pose estimate to $F_{A,i+1}$. This is done by HighlyAccurate [33]. All four methods retrieve a 3-Degree of Freedom (DoF) camera pose.

The disadvantages of the hypothesis formulation is that there is an irreducible error that comes from the fact that hypotheses are discrete. The disadvantage of the iterative pose refinement, as opposed to the hypothesis formulation, is that it is not guaranteed to converge to the global optimum, although Levenberg-Marquardt (LM) is guaranteed to find the local one. Furthermore, the iterative pose refinement cannot be parallelized, thus is computationally expensive.

The influence of the number of hypotheses on the performance is difficult to establish, since none of the works conduct an ablation study on this parameter. At test time SliceMatch uses 15×15 location hypotheses and 64 for orientation. During training, the values are lower, but the ratio is almost identical. CCVPE [35] uses dense classification for position, i.e. for each of 512×512 pixels and 20 orientation bins. Clearly, in relative terms, the SliceMatch attributes more of its hypotheses to orientation, whereas CCVPE attributes more to position. Interestingly, the results on Table 2-3 do not reflect this, since on the ± 10 degree orientation setting SliceMatch is more similar to CCVPE in terms of localization performance rather than orientation. It can thus be concluded that the underlying methodology has more influence on the results than the density of hypotheses.

2-3-5 Quality of the Ground Truth

Many works, e.g. [39, 33] report the quality of the ground truth to be one of the most severe limitations in reaching better performance. HighlyAccurate [33] improve the ground truths by manual filtering to create a subset of only 19 655 filtered images. When training on this subset as opposed to the full dataset, the lateral recall at 1 is improved from 26.67 % to 46.10 %.

Table 2-5: Inference speed for CVGL methods. Note that the hardware used is not always the same. For instance, UncertaintyAware was evaluated with RTX 6000 whereas SliceMatch and CCVPE with Tesla V100, which is a slower GPU. Also note that inference speed depends on the dataset and not all methods were evaluated on the same one.

	CVR [81]	Accurate3DoF [82]	MCC [38]	HighlyAccurate [33]	UncertaintyAware [39]	SliceMatch [34]	CCVPE [35]
Speed [ms]	20	500	330	500	420	6	70

Similarly, after identifying the quality of the ground truth to be a limiting factor, UncertaintyAware [39] propose a way to improve the ground truth of existing datasets by aligning aerial images with the LiDAR point cloud. More information is given in subsection A-2-2. The ablation study shows that without the ground truth refinement, the mean error on the predicted camera pose is almost exactly twice larger, which is extremely significant.

Methodologically, the improving the ground truths the way it was done by UncertaintyAware is likely more robust than manual filtering, performed by HighlyAccurate. The resulting ground truth quality could still be a large contributor to the superior performance of UncertaintyAware over HighlyAccurate.

More information about existing datasets is given in section 2-4.

2-3-6 Time Complexity

Having identified the localization speed requirements in subsection 2-1-3, we now evaluate the speed of existing CVGL methods. The reported inference time is shown on Table 2-5.

There is another expected relationship between the methods and their runtimes. The methods that compare entire feature maps, i.e. HighlyAccurate [33] and UncertaintyAware [39] are 1-2 orders of magnitude slower than the methods that compare local features, i.e. SliceMatch [34] and CCVPE [35]. This is another advantage of using local features at the hypothesis level.

Generally, the runtimes of the local feature-based approaches are already similar to the requirements in subsection 2-1-3. There the localization rate requirement of 5 Hz was identified, which translates to a 200 ms inference speed. SliceMatch and CCVPE already satisfy this requirement.

2-3-7 Lack of Interpretability

Some CVPE methods discussed in this section possess a degree of interpretability. Most do so only at the camera pose level, providing not only the retrieved location, but a confidence map, which typically results from evaluating each hypothesis. By comparing confidence maps between different images it is possible to deduce what the model finds important and uses to make predictions. For instance, on Figure 2-8, it seems that because of the tree directly in front of the vehicle on the ground image, the hypotheses which would contain the tree at the correct location are more likely. However, this is merely an assumption. It could be the road markings or the building layout, which guide the model to the correct location.

SliceMatch [34] is unique is that due to the use of local features, the authors are also able to recover attention maps from the attention module that they use. Attention maps provide a

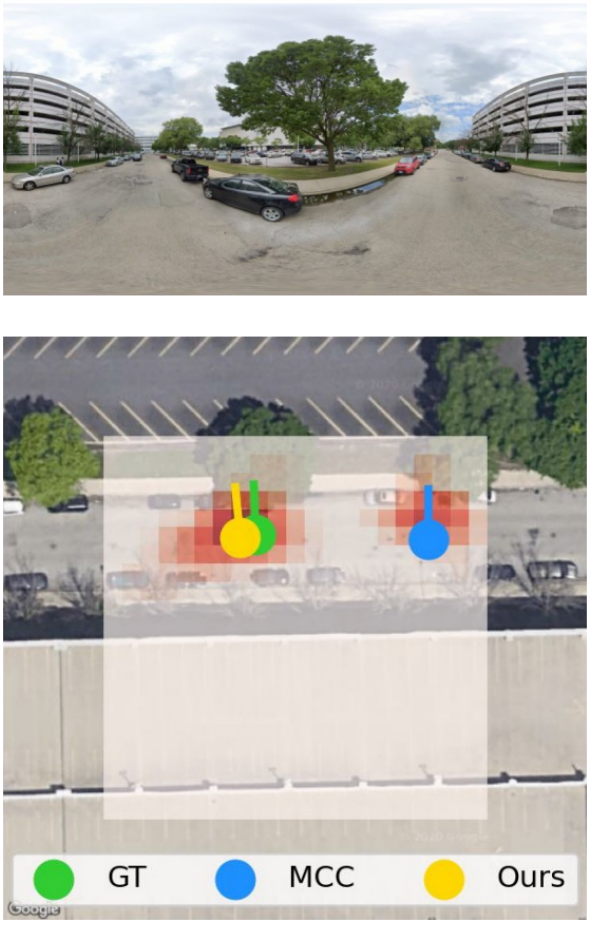


Figure 2-8: Confidence map example from SliceMatch [34]. Similar confidence maps are obtained by other methods that use a hypothesis formulation, e.g. [39, 35].

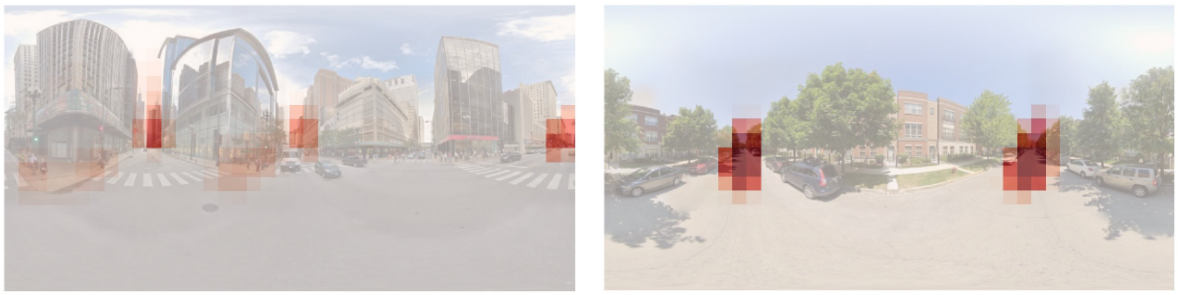


Figure 2-9: Attention map examples from SliceMatch [34].

larger degree of interpretability - using Figure 2-9 it is possible to see whether it is the road markings, the building layout or the tree that the model uses to make predictions.

Nevertheless, the areas that are important for descriptor generation can not explicitly explain why a certain hypothesis is more likely. Improving interpretability is important not as much for a performance improvement on the test set, as for the ability to understand the model's behaviour, which might reveal insight into the problem itself.

2-3-8 Evaluating the Localization Setting

As we mentioned in section 2-2, the first visual CVGL works formulated the task as image retrieval on a dataset with centered aerial images. Datasets adopted by CVIR methods [7, 83, 84], such as CVACT, CVUSA and Oxford Robot Car contain aerial images that typically span more than $50\,\text{m} \times 50\,\text{m}$. For instance, [7] collected the aerial images for the Oxford Robot Car dataset, which cover $55.4\,\text{m} \times 55.4\,\text{m}$ and [84], who introduce CVACT, also present a localization architecture, where images covering a $144\,\text{m} \times 144\,\text{m}$ area are used.

Such coverage of an aerial image was likely motivated in retrieval works by the fact that it corresponded best to the semantics present in the ground image. It is likely an optimal size of the aerial image to both retain tall objects far away visible in the ground view, yet also capture objects close by with a reasonable resolution. The coverage was thus chosen to yield the best accuracy for the retrieval task.

Afterward, the task was formulated as CVPE by [81] and [82]. Both works adopted a two-stage approach, where the image is first retrieved and then pose estimation is conducted within the retrieved image. The follow-up works in the field [33, 38, 85, 34, 39], however, claimed that since the local aerial patch can be retrieved through any rough localization prior, e.g. GPS/GNSS or temporal filtering, the image retrieval objective can be dropped. The works thus focus on the second stage of the pipeline only, i.e. camera pose estimation, given the correct aerial patch.

While such a formulation has an indubitable motivation, it raises the question about the expected quality of the prior. Some works [85, 38] consider the entire aerial patch on the VIGOR dataset to constitute the possible vehicle location. The remaining methods pre-define the prior manually, which is smaller than the aerial patch, yet still quite uncertain, typically $20\,\text{m} \times 20\,\text{m}$ for position and $10°$ for orientation.

Such uncertainty follows from the fact that the aerial image needs to extend far enough to capture semantic information about the scene, regardless of the orientation of the vehicle,

thus the vehicle must be located within the central region of the aerial patch. However, this motivation for the prior is linked to the size of the aerial patch developed for CVIR methods rather than stemming from a feasible localization prior from GNSS, temporal filtering or global image retrieval methods. Therefore, there is room for improvement in the localization setting itself to bridge the gap with the industry requirements.

2-4 Relevant Datasets and Metrics

In this section the relevant datasets and evaluation metrics are discussed. The datasets are discussed in subsection 2-4-1 and the metrics in subsection 2-4-2. Only CVPE works are considered, since retrieval works are deemed not relevant for the fine-grained localization task.

2-4-1 Metric CVPE Datasets

There are a number of datasets that have been used for metric CVPE. These contain more accurate ground truth poses, compared to those in the previous section. The ground truths were extracted from survey-grade GPS-RTK systems, which are significantly more accurate than raw GPS (e.g. Oxford RobotCar [86, 87], KITTI [70], KITTI-360 [88]) and additionally post-processed by synchronizing with velodyne data (e.g. KITTI [70], KITTI-360 [88]).

The overview of the datasets is presented on Table 2-6. Note that the authors of both [33] and [39] noticed a lot of inaccurate poses in the datasets they used, even when highly accurate RTK data is used as the ground truth. Both of these works proposed a method to filter out the inaccurate poses. The authors of [33] simply manually removed images that visually did not correspond to their aerial ground truths. The authors of [39] used LiDAR data to align ground truths with the aerial data and retrieved new ground truths, as shown on Figure A-5.

The first three datasets on Table 2-6 are readily available for use. The remaining datasets are not publicly available and will not be released due to licensing issues. Because of the large mismatch in the size of each of the first three datasets and the ones used by UncertaintyAware [39], the results of the latter are not directly comparable to other methodologies.

For this thesis work, the processed KITTI dataset will be used, as it contains accurate 6DoF ground truth poses and has been used extensively in the literature. Furthermore, the ORC dataset was used to additionally evaluate the best performing model, though quantitative results are not reported.

2-4-2 CVPE Metrics

For CVPE, the most commonly used metrics are the mean and median position and orientation errors per query. When methods use a hypothesis formulation, e.g. [34, 39], retrieval results are also reported. Retrieval in this case refers to retrieving the correct hypothesis within the image rather than retrieving the correct match.

Furthermore, when probabilistic filtering is also used, the authors additionally report the mean and median error of the filtered trajectory and the estimated probability at the ground truth, e.g. [38, 39].

Table 2-6: Overview of datasets used in CVPE literature.

Dataset	Source	Number image pairs	Ground image	Year	Used by
Processed KITTI	[70] (aerial data from [33]) (processed by [33])	30 970	Limited FoV	2013	[33, 34, 35]
ORC	[86] (ground truths from [87], aerial data from [8])	23 854	Panorama	2017	[35, 38]
Processed Ford AV	[69] (aerial data from [33]) (processed by [33])	50 851	Panorama	2017	[81, 34, 33]
Ford AV	[69] (aerial data from [39])	136 000	Panorama	2017	[39]
Argoverse V1	[89] (aerial data from [39])	22 000	Panorama	2019	[39]
Argoverse V2	[90] (aerial data from [39])	844 000	Panorama	2021	[39]
Lyft L5	[91] (aerial data from [39])	50 000	Panorama	2019	[39]
KITTI-360	[88] (aerial data from [39])	76 000	Panorama	2022	[39]
nuScenes	[92] (aerial data from [39])	19 000	Panorama	2018	[39]
Pandaset	[93] (aerial data from [39])	8000	Panorama	2019	[39]

For the quantitative results of the thesis work we will report the mean and median position and orientation errors per query, as well as the retrieval results. No temporal filtering will be used.

2-5 Limitations of Existing Works and Research Directions

Overall, there are six key limitations related to the CVPE field that can be used to guide the intuition for the subsequent research directions:

1. There is a mismatch between the existing localization setting and the one required by the industry. A real-life vehicle is able to ensure a decently accurate localization prior using GNSS for instance, so assuming a $20\,\text{m} \times 20\,\text{m}$ uncertainty region on the camera pose is unjustified. On the other hand, the required localization performance needs to be improved significantly to match the industry standard.

2. When trained on publicly available datasets, the state-of-the-art CVGL methods demonstrate generally poor performance on the cross-area setting, some close to random. Moreover, it is likely that the road prior is the most important factor for any incremental improvements in performance.

3. Most methods use local features. Local features generally result in better performance in terms of both accuracy and speed. Furthermore, local features are more interpretable than global features, which is important for understanding the model's behaviour. Nevertheless, when supervision occurs at the camera pose level, interpretability is limited, since the effect of local matches on the camera pose is latent.

4. Existing works, most of which use local features with image-level supervision or global features entirely, are not data efficient, which is a limitation, given the relatively small sizes of existing, publicly available datasets.

5. Methods that use iterative pose refinement instead of the hypothesis formulation perform worse. This is due to the fact that the iterative pose refinement is not guaranteed to converge to the global optimum as well as the slow inference speed.

6. Longitudinal performance of the state-of-the-art is significantly worse than lateral. Due to the use datasets with only forward facing cameras in all but one methods, the longitudinal performance is largely dependant on the implicit depth prediction capacity of the backbone, which is typically poor with respect to the accuracy requirements.

Using the limitations outlined above, we first propose a novel localization setting, which bridges the gap between what is currently used in the CVPE field and what is required by the industry, addressing Limitation 1. We call this setting the "fine" localization setting. Afterwards, we propose three novel approaches to the fine-grained localization problem: Regression, Point-to-Point (P2P) matching and Ray-to-Point (R2P) matching.

Firstly, by analyzing the failure cases of HighlyAccurate [33] in Figure A-3, particularly the influence of multi-modal features or those with uninformative derivatives, we discover

that those are particularly detrimental in the coarse localization setting. In the more fine grained setting, we hypothesize that these failure cases will not be as limiting, since the overlap between feature maps will be larger. We thus propose a method largely based on HighlyAccurate, with some modifications, motivated by the conclusions of subsection 2-3-4 and Appendix B. Notably, we replace the iterative pose refinement with a single pose regression step, addressing Limitation 5. We call this method the **Regression method**.

Secondly, due to the numerous advantages of using local features identified in Limitation 3 we propose two methods that predict the camera pose by matching the most fine-grained local features attempted so far. We hypothesize that using "very local" features is well suited for fine-grained localization. Inspired by the recent advancements in the field of transformers, we recognize that the attention mechanism happens to be well suited to predicting the locations of query ground features based on the reference aerial keys. Furthermore, due to the issues with interpretability and dataset size (Limitation 3 and Limitation 4), we choose to supervise the model at the local feature level rather than on the camera pose level to make the model both more interpretable and data efficient. We call these two methods **Local Feature Matching methods**.

The first local feature matching method, **P2P** initializes a ground and an aerial point clouds, which contain features from respective images. The ground point cloud is initialized in the vehicle frame, while the aerial point cloud is initialized in the global, world reference frame. Cross-attention is applied on the ground points to predict their correct locations in the global reference frame. Using a set of point correspondences, the camera pose is predicted using either a hypothesis formulation or iterative pose refinement.

The second local feature matching method, **R2P** is similar to P2P, except for the fact that it also aims to address Limitation 6. Instead of initializing ground points, in R2P ground rays are used. This rids the ground feature extractor of the need to estimate depth, which means that the cross-attention matching is based on the semantic content only and thus is not dependant on the implicit depth prediction capacity of the backbone. The camera pose is predicted to minimize the total angular error between the ground rays and the "rays" that the predicted aerial matches make with the camera pose.

Finally, by using a projective transform we discover that even the processed KITTI dataset used by [33, 34, 35] still contains inaccurate ground truth poses on both the large scale (clearly obvious errors) and small scale (those that are difficult to pinpoint visually).

2-6 Thesis' Contributions

The contributions of the thesis are as follows:

1. We propose two local feature matching methods, which are uniquely supervised on the local feature level. The methods are:
    - More interpretable than current CVPE methods.
    - Uniquely able to predict a 6-DoF camera pose.
    - More data efficient than current CVPE methods.

2. We address the shortcomings of HighlyAccurate [33] and tailor it to the new localization setting.
3. We propose and motivate the need for a novel localization setting, which is more fine-grained than what currently available methods target.
4. We discover both fine-grained and large-scale errors in the ground truth of the processed KITTI dataset.

Chapter 3

Methodology

This chapter describes the methodology used to answer the research questions defined in section 1-4. We first establish the novel localization setting in section 3-1. Afterwards we define the problem formally and introduce the data augmentation setup used ubiquitously in Cross-View Pose Estimation (CVPE) literature [34, 35, 33] in section 3-2. The baseline approach is defined in section 3-3. In section 3-4 we briefly introduce and motivate the three approaches that we propose to address the research questions: Regression, Point-to-Point (P2P) and Ray-to-Point (R2P) matching.

We describe each of the three approaches in detail, in section 3-5, section 3-6 and section 3-7 respectively. Afterwards, we present an overview of the implementation details for the sake of reproducibility in section 3-9.

In section 3-5, section 3-6 and section 3-7 we also define a number of hypotheses that are based on the research questions defined in section 1-4, yet specific to the methodology in each approach. These hypotheses are summarized in section 3-10 and are used to guide experiment design in chapter 4.

3-1 Novel Localization Setting

One of the conclusions of chapter 2 is a mismatch between the localization setting widely used in CVPE literature and the required localization setting defined by industry demand. In this section we define the novel localization setting, which is used throughout this work.

The localization setting is defined by the uncertainty on the camera pose, which stems from the best available prior. As we concluded in section 2-3, the best visual localization methods already use a prior, since they do not perform image retrieval themselves. Therefore, we envision a three-stage purely vision-based localization pipeline as shown on Figure 3-1, where the fine-grained localization algorithm is one of the thesis' contributions.

In order to build an architecture with a functionality of the green box in Figure 3-1, we need to know the quality of the prior provided by either option for the medium localization estimate.

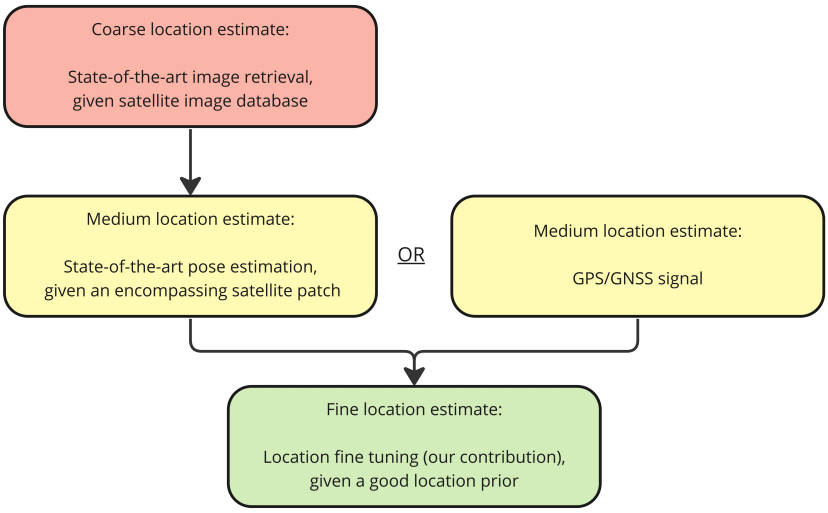


Figure 3-1: Proposed localization pipeline.

Table 3-1: Example of localization performance to provide a medium localization estimate. Gray rows correspond to the anticipated use case. From SliceMatch [34]

(a) KITTI dataset

Area	Prior	Location error Mean [m]	Location error Median [m]	Lat. recall (%) r@1 m	Lat. recall (%) r@5 m	Long. recall (%) r@1 m	Long. recall (%) r@5 m
Same	20°	7.96	4.39	49.09	98.52	15.19	57.35
Same	✗	9.39	5.41	39.37	87.92	13.63	49.22
Cross	20°	13.50	9.77	32.43	86.44	8.30	35.57
Cross		14.85	11.85	24.00	72.89	7.17	33.12

(b) VIGOR dataset

Area	Aligned images	Location error Mean [m]	Location error Median [m]
Same	✓	5.18	2.58
Same	✗	8.41	5.07
Cross	✓	5.53	2.55
Cross	✗	8.48	5.64

If we design a purely vision-based localization pipeline in the absence of the GPS signal, the medium localization estimate is provided by the state-of-the-art CVPE methods, outlined in section 2-3. Because the localization algorithm should be applicable to real world settings, we concur with [39], who claim that because same-area evaluation protocol requires obtaining data from the target region in advance, it "contradicts our motivation for using aerial images, i.e. global availability at low cost". We make the same claim about aligned images, arguing that in real world settings the images are not aligned. Regarding the angular prior, however, we believe that it is feasible that a 20° prior is available in advance. An example of the state-of-the-art method evaluated in this setting is SliceMatch, whose performance is encapsulated by the gray rows of Table 3-1.

The gap between the target localization performance (decimeter level accuracy at 95% confidence) and the current state-of-the-art shown on Table 3-1 is enormous. Moreover, if the medium localization estimate is provided by the GNSS, Table 1-1 demonstrates that the localization performance is not significantly better with ±5 m errors due to the atmospheric effects alone.

In order to make the fine-grained localization task feasible, we assume a non-conservative medium localization estimate with 5 m position uncertainty. In terms of the orientation prior, we propose to maintain the uncertainty of 10°. The reason is that GPS receivers do not

provide location information, while state-of-the-art medium-level pose estimation methods are not able to provide better orientation estimates when not fed with any orientation prior themselves, as shown on Table 2-3.

3-2 Problem Definition

The objective of CVPE is to determine the camera pose X_c in the aerial image $I_a \in \mathbb{R}^{H_a \times W_a \times 3}$ using the ground image $I_g \in \mathbb{R}^{H_g \times W_g \times 3}$, where H_a, W_a, H_g, W_g are the height and width of the aerial and ground images respectively. In this work we consider both a 3-DoF camera pose $X_c = [x_c, y_c, \psi_c] \in \mathbb{R}^3$ and a 6-DoF camera pose $X_c = [x_c, y_c, z_c, \phi_c, \theta_c, \psi_c] \in \mathbb{R}^6$, where x_c, y_c, z_c are the camera's position in the world frame, ϕ_c, θ_c, ψ_c are the camera's roll, pitch and yaw respectively.

At both train and test time we generate the ground truth x_c, y_c, ψ_c from a uniform distribution. The longitudinal and lateral pixel offsets are generated on the interval $[-x_{max}, x_{max}]$ and $[-y_{max}, y_{max}]$ respectively, while the yaw angle is generated on the interval $[-\psi_{max}, \psi_{max}]$. In the proposed "fine" localization setting we use $x_{max} = y_{max} = 5\,\text{m}$ and $\psi_{max} = 10°$.

Using the generated x_c, y_c, ψ_c, the aerial image I_a is manipulated such that it is centered at a point $(-x_c, -y_c)$, which is x_c meters away from the ground truth in the direction of true vehicle motion (longitudinally) and y_c meters away laterally. Furthermore, the aerial patch is rotated such that the angular difference between the east direction on the aerial image and the true heading is $-\psi_c$. Since the aerial image is geo-referenced, predicting X_c offsets is equivalent to predicting the camera pose in the global reference frame.

3-3 Baseline

Due to the novelty of the proposed localization setting, only a few works have attempted to solve it. In fact, as we have concluded in chapter 2, to date there are no camera-only works that attempt to solve the problem of fine-grained CVPE. To the best of our knowledge the only method that has been designed specifically for cross-view fine-grained localization involves the use of a LiDAR sensor [94].

The authors of [94] compare the proposed approach to HighlyAccurate [33], which has been trained and evaluated on the fine-grained localization task, defined similarly to what we have defined in section 3-1. The only difference is that the orientation uncertainty in [94] was set to 15° instead of 10°, as is the case in this work.

Therefore, in this thesis work **HighlyAccurate [33] is also used as the baseline for the fine-grained setting**. We re-train HighlyAccurate in the setting defined in section 3-1, namely such that the position uncertainty is 5 m and the orientation uncertainty is 10°. An in-depth overview of the methodology and evaluation is given in section A-1.

3-4 Overview of Proposed Approaches

For the problem formulation defined in section 3-2, we explore three approaches:

1. **Regression.** A regression-based approach, which uses a projective transform, similar to HighlyAccurate [33], to bridge the domain gap between the aerial and ground images.
2. **P2P matching**. A local feature matching approach, which uses a set of aerial and ground points to predict the camera pose.
3. **R2P matching**. A local feature matching approach, which uses a set of aerial points and ground rays to predict the camera pose.

Below each approach is motivated and briefly introduce. More detailed explanation is provided in section 3-5, section 3-6 and section 3-7 respectively.

3-4-1 Regression Approach Introduction

One of the conclusions of the the related works, identified in section 2-2 is the need to explicitly bridge the domain gap between aerial and ground images. Most Cross-View Image Retrieval (CVIR) methods do so by either using an explicit, closed-form, geometric transformation, post-processing the features or using local features. For the regression approach, we decided to use a projective transform, similar to HighlyAccurate [33] to bridge the domain gap on either the original images or their feature maps. We then concatenate image descriptors and feed them into an Multilayer Perceptron (MLP) to regress the camera pose.

An example regression approach is depicted on Figure 3-2. The architecture is similar to HighlyAccurate [33], with three key exceptions: (1) a single pose regression step is performed instead of iterative refinement, (2) multi-scale feature comparison of HighlyAccurate is replaced with a single-scale comparison of features extracted via a ResUNet backbone, (3) 6-DoF pose is predicted instead of 3-DoF. Therefore, first either the ground or the aerial image is projected across the domain gap using the projective transform, as described in

the features are extracted using a backbone, then they are flattened, concatenated and fed through an MLP to determine the 6-DoF camera pose, as shown on Figure 3-2.

3-4-2 P2P Matching Approach Introduction

Despite the novel, 6-DoF output of the regression approach, it still uses global feature maps. Local features, on the other hand, have been linked to superior performance and interpretability in chapter 2 as well as intuitive advantages in data efficiency. Therefore, we propose a local feature matching approach, which uses a set of aerial and ground points to predict the camera pose.

The architecture of the P2P matching approach is shown on Figure 3-3. The goal is to have two sets of points, one containing features from the aerial image and the other containing features from the ground image. For each ground point, point cross attention is used to determine the highest likelihood location of the query ground point in the aerial point cloud. This way we obtain a set of target locations for a set of geo-referenced ground points. Afterwards, we use a bundle adjustment-like approach to recover the camera pose, given the set of point correspondences.

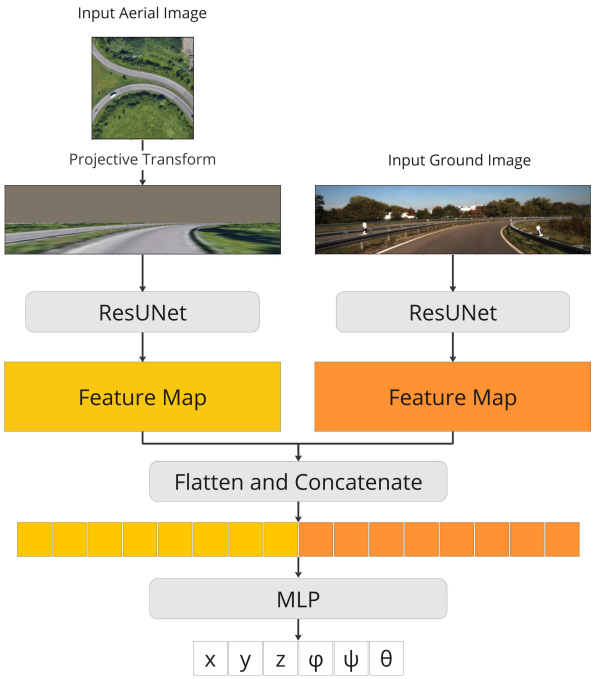


Figure 3-2: Example of a regression approach. Alternatively, the ground image may be projected onto the aerial domain or the projection module may be absent altogether. The order of feature extraction and projection may also be swapped.

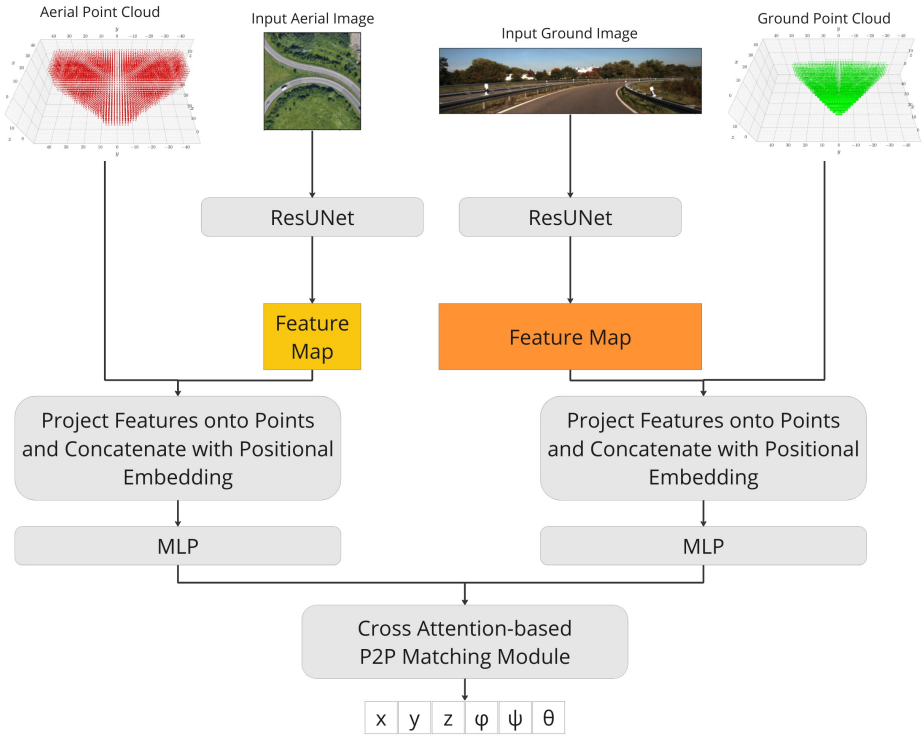


Figure 3-3: Overview of the P2P matching approach.

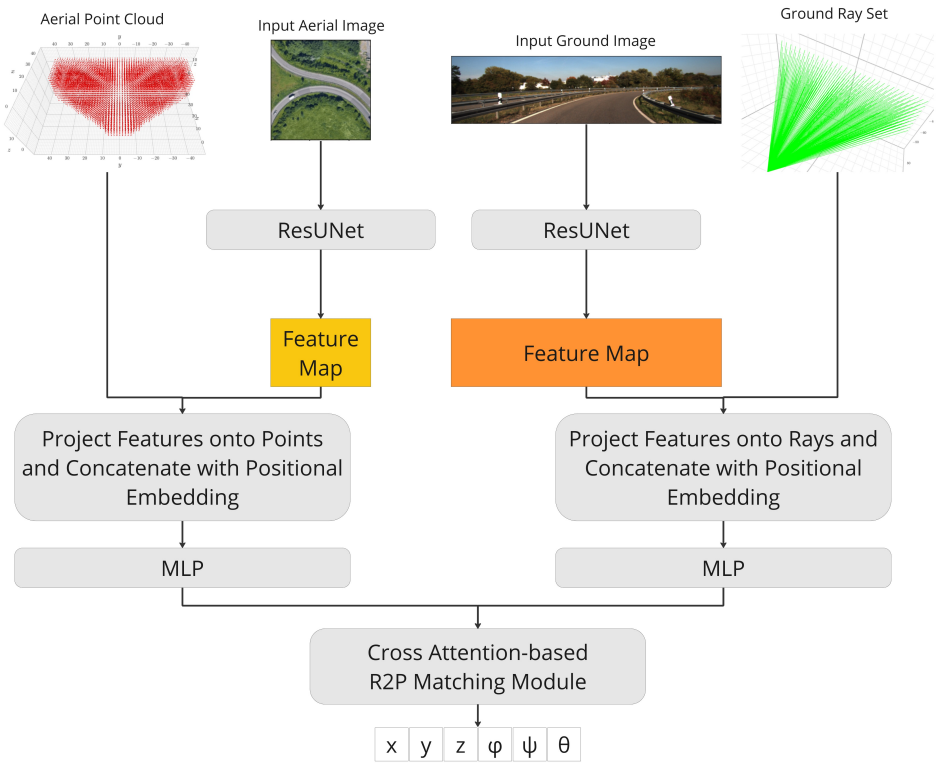


Figure 3-4: Overview of the R2P matching approach.

3-4-3 R2P Matching Approach Introduction

While the P2P matching approach offers a highly interpretable output, such that the match for each local feature on the ground can be visualized, the approach requires all points in the ground point cloud to be "matchable" with some scene content in the aerial image. Due to occlusions and empty space, this is not always the case, as will be further discussed in chapter 4.

Each pixel on the ground image corresponds to a ray in 3D space, originating at the camera location. In fact, for each such ray, there is at most one distinct "matchable" aerial point p'_a. All ground points that lie on the ray, but are closer to the camera than p'_a would correspond to empty space. All ground points that lie on the ray, but are farther away from the camera than p'_a are occluded by the object located at p'_a.

Therefore, instead of predicting matches for each ground point, in R2P matching we propose to predict matches for each ground ray via an otherwise similar approach. The architecture of the R2P matching approach is shown on Figure 3-4.

3-5 Regression Approach

The main component of the regression approach is the 6-Degree of Freedom (DoF) projection transformation module, which is discussed in detail in subsection 3-5-2. Prior to this, however, we explore the different possibilities regarding the order of components on Figure 3-2. We

can, for instance, apply the projective transform before or after extracting features and on either/neither of the two input images. This is discussed in subsection 3-5-1.

3-5-1 Architecture Variations

The first two steps of the algorithm is to bridge the domain gap between the ground image I_g and the aerial image I_a and extract image descriptors F_g and F_a, respectively. These steps can be performed in any order. In this work we consider two options: (1) project the original images and then extract features, (2) extract features and then project the features.

Our first hypothesis **AH1** is that projecting feature maps F_g and F_a is more effective than projecting the original images I_g and I_a because the perspective projection is not a one-to-one transformation, thus some information is lost. Extracting features first gives the network the freedom to learn which information is important and which is not.

In terms of the projective transformation module, we consider three options: (1) project the aerial image onto the ground domain, (2) project the ground image onto the aerial domain, (3) do not project at all. Our second hypothesis **AH2** is that option (1) will perform slightly better than option (2) and much better than option (3). This is because of the importance of a domain bridging module in the CVIR field as discussed in section 2-5 and the predominant number of works that do so in the ground domain.

3-5-2 Projective Transformation

In order to bridge the domain gap between the aerial and ground views, we use the projective transformation with the homography assumption, identical to the one used by HighlyAccurate [33].

The transformation uses camera intrinsics to project the pixel intensities between the ground and aerial views. We consider three cases: (1) where the aerial image is projected onto the ground domain, (2) where the ground image is projected onto the aerial domain, (3) where no projection is performed.

Suppose the relationship between a ground image pixel $[x_g, y_g]$ and the world reference coordinates $[X, Y, Z]$ is given by Equation 3-1 and the relationship between an aerial image pixel $[x_a, y_a]$ and the world reference coordinates $[X, Y, Z]$ is given by Equation 3-2.

Then the relationship between a ground image pixel $[x_g, y_g]$ and an aerial image pixel $[x_a, y_a]$ is given by Equation 3-3.

$$s \begin{bmatrix} x_g \\ y_g \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\psi_c & -\sin\psi_c & 0 & x_c \\ \sin\psi_c & \cos\psi_c & 0 & y_c \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3-1}$$

$$\begin{bmatrix} X \\ Y \end{bmatrix} = A \begin{bmatrix} x_a \\ y_a \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_a \\ y_a \end{bmatrix} \tag{3-2}$$

$$s\begin{bmatrix} x_g \\ y_g \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\psi_c & -\sin\psi_c & 0 & x_c \\ \sin\psi_c & \cos\psi_c & 0 & y_c \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & 0 & 1 \\ a_{21} & a_{22} & 0 & 1 \end{bmatrix} \begin{bmatrix} x_a \\ y_a \\ 0 \\ 1 \end{bmatrix} \tag{3-3}$$

where f_x, f_y are the focal lengths, c_x, c_y are the principal points, x_c, y_c, z_c are the camera's position in the world frame, ψ_c is the camera's yaw angle, X, Y, Z are the world reference coordinates, x_g, y_g are the ground image pixel coordinates, x_a, y_a are the aerial image pixel coordinates, $a_{11}, a_{12}, a_{21}, a_{22}$ are the elements of the affine transformation matrix A, which transforms the aerial image coordinates into the world reference frame. Finally, s is the depth of the pixel in the world reference frame.

The homography assumption is present in the fact that the affine transformation A is 2-dimensional. All aerial image pixels are projected onto the ground plane with height 0.

The transformation shown in Equation 3-3 can be sped up by pre-computing constant terms. $[x_a, y_a]$ is a constant set of pixel coordinates that should be transformed at each inference step. Furthermore, affine extrinsic matrix A as well as the intrinsic matrix are constant. Also, the depth s does not depend on camera extrinsics, thus can also be pre-computed. During this pre-computation we also create a mask for the depth map s such that points behind the camera are not projected.

After projecting the aerial image onto the ground domain, we interpolate the sparse set of features/RGB values to obtain a dense feature map/RGB image of the required size. We use bilinear interpolation for this purpose. Equations above can also be inversed to project the ground image onto the aerial domain.

However, in fine grained settings we hypothesize that the effect of pitch, roll and altitude of the camera can no longer be neglected, as is the case throughout Cross-View Geo-Localization (CVGL) literature. Therefore, we modify the projective transform to include pitch and roll. Unfortunately, no altitude information is available in any of the datasets listed in subsection 2-4-1, therefore the zero-altitude assumption is maintained.

The camera updated extrinsic matrix is given in Equation 3-4 and is substituted in Equation 3-3.

$$E = \begin{bmatrix} \cos\psi_c & -\sin\psi_c & 0 & x_c \\ \sin\psi_c & \cos\psi_c & 0 & y_c \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_c & 0 & \sin\theta_c & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_c & 0 & \cos\theta_c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi_c & -\sin\phi_c & 0 \\ 0 & \sin\phi_c & \cos\phi_c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3-4}$$

3-6 P2P Matching

The P2P matching architecture is shown on Figure 3-3. First of all, the ground and aerial point clouds are initialized in 3D space. This is done only once at the beginning of the training/inference sequence. More details on how the points are initialized and why some of them are discarded are given in subsection 3-6-1 and subsection 3-6-2 respectively. Furthermore,

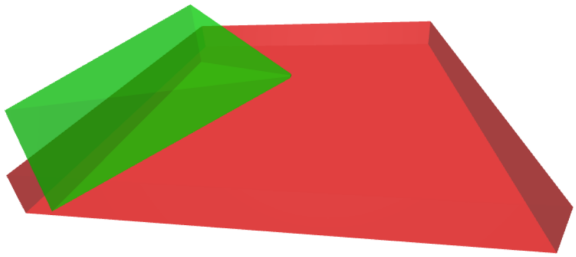


Figure 3-5: The aerial (red) and ground (green) point spaces.

the look-up pixels on the ground image are pre-computed based on the ground point cloud and the camera intrinsics.

During each forward pass, the ResUNet architecture is used to extract features from the ground and aerial images, as described in subsection 3-9-1. Using the look-up pixels, the relevant image features are attributed to each point in the point cloud. Furthermore, the positional information of each point is added to its feature vector, as described in subsection 3-6-3. The positional and semantic information is merged together using a single MLP layer to create point tokens, as discussed in subsection 3-6-4. This is a standard step in Vision Transformer (ViT) architectures [95].

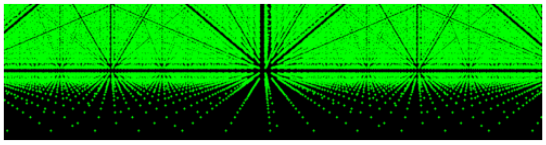
Afterwards, in subsection 3-6-5 we describe the key contribution of our work - the point cross attention module. It is responsible for predicting the location of each token associated with a ground point using the aerial point cloud. In its basic form, the attention module treats the ground points as queries, the aerial points as keys and the locations of the aerial points as values. More details are provided in subsection 3-6-5.

Finally, the loss function, which is based on individual point losses, is described in subsection 3-6-6. Two options for the loss are provided, the Mean Squared Error (MSE) loss and the Negative Log-Likelihood (NLL) loss, which additionally takes into account the uncertainty of the prediction.

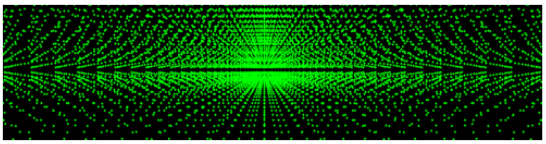
3-6-1 Generating Points

The first step is generate a set of aerial and ground points in the world coordinate frame. Aerial points are generated on a uniform grid inside a cuboid inscribing the satellite patch. For the KITTI dataset this cuboid spans $100 \times 100\,\text{m}$. The ground points are generated on a uniform grid inside a frustum inscribing the ground-level image. The length of the frustum is limited by the satellite cuboid, see Figure 3-5.

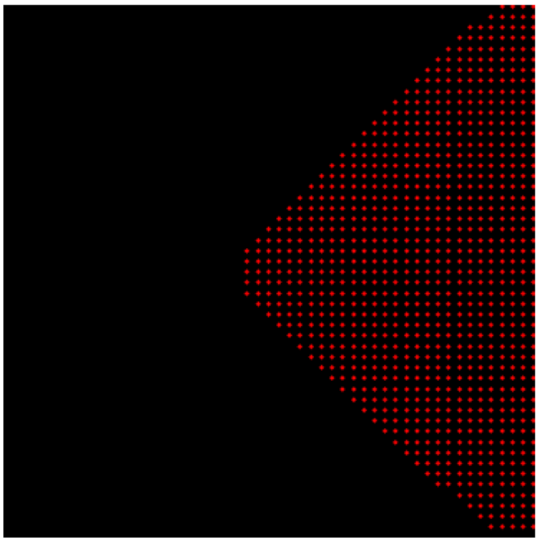
The set of 3D ground points $P_g = \{p_g^1, ..., p_g^{N_g}\},\ p_g^i \in \mathbb{R}^3\ \forall\ 1 \leq i \leq N_g$ are then sampled within the ground space and the set of 3D satellite points $P_a = \{p_a^1, ..., p_a^{N_a}\},\ p_a^i \in \mathbb{R}^3\ \forall\ 1 \leq i \leq N_a$ are sampled within the satellite space. Sampling is performed either uniformly or exponentially. The latter can be done to ensure that more information is recovered from the satellite and ground images. See Figure 3-6 for an illustration.



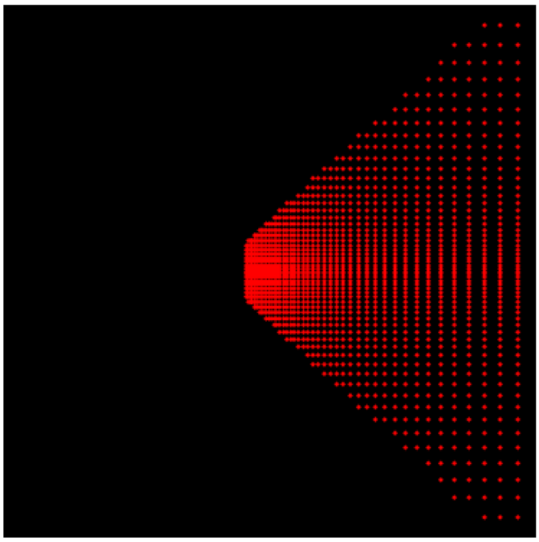
(a) Uniform sampling in 3D, projected back to ground



(b) Exponential sampling in 3D, projected back to ground



(c) Uniform sampling in 3D, projected back to aerial



(d) Exponential sampling in 3D, projected back to aerial

Figure 3-6: Projecting sampled points back to the ground and satellite images. The left column has approximately twice as many points as the right column. Each pixel is magnified by a factor of 4 for visualization purposes.

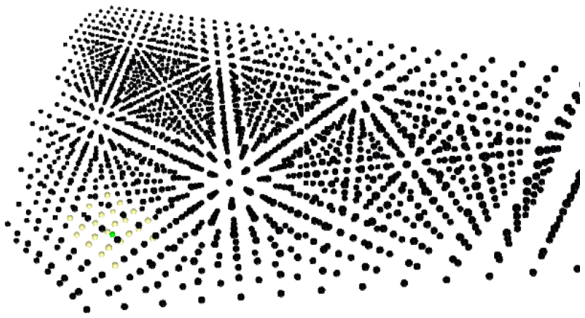


Figure 3-7: The matchable (yellow) and unmatchable (black) aerial points to a ground query (green).

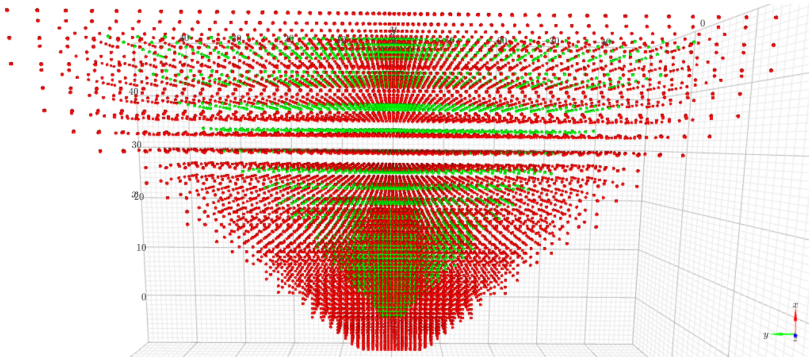


Figure 3-8: The pruned aerial points (red) and ground points (green).

3-6-2 Pruning Aerial Points

The subsequent localization pipeline involves locating a ground point by attending to pose-referenced satellite points. Because there is a strong prior on vehicle's position, each p_g can be associated with the subset of P_a which should be sufficient to localize it. This corresponding subset of P_a is a function of the 3D ground point p_g and the maximum offsets: x_{max}, y_{max} and ψ_{max}, as well as the maximum pitch and roll angles θ_{max}, ϕ_{max}. Note that maximum pitch and roll angles are hyperparameters, which largely depend on the dataset.

After computing subsets of P_a for each p_g, we construct a mask matrix $M^{N_g \times N_a}$ such that $M_{i,j} = 1$ if p_g^i is associated with p_a^j and $M_{i,j} = 0$ otherwise. The mask matrix M is then used to prune the set of aerial points P_a to $P_{a,p} = \{p_{a,p}^1, ..., p_{a,p}^{N_{a,p}}\}$, where $N_{a,p} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_a} M_{i,j}$ and $p_{a,p}^i \in P_a \; \forall \; 1 \leq i \leq N_{a,p}$. An example of the subset of aerial points associated with a ground point is shown in Figure 3-7. The final pruned points are shown on Figure 3-8. These points remain fixed throughout the training process, although there are clear opportunities for regenerating points based on their influence on the localization error, similar to a particle filter.

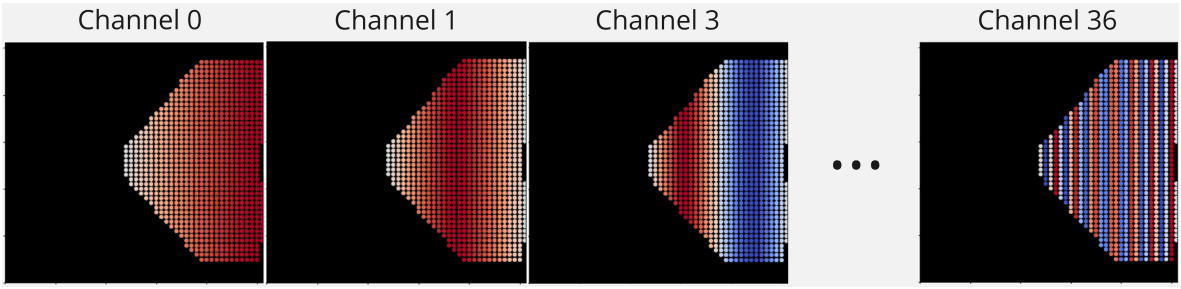


Figure 3-9: Example of positional embedding on the longitudinal coordinate of the aerial points. The first channel is discriminative on the largest scale, while the last channel is discriminative on the smallest scale.

3-6-3 Positional Encoding

There are two reasons for the use of positional information in the point descriptors. First of all, it provides content, which may improve the quality of the descriptors. For instance, if a local feature resembles road markings, but is located at the top of the image, the positional information will allow the model to refine descriptor accordingly, reducing the probability that it is indeed a road marking. Secondly, if two points lay along the same ray in the camera view or along the height dimension in the aerial view, they will have identical descriptors and yet need to be distinguished by the model. Depth and height embedding respectively makes this possible. A number of methods have been attempted to encode positional information into point descriptors. A number of experiments have been conducted to determine the best performing method, which turned out to be the sinusoidal positional embedding.

Sinusoidal positional encoding. Sinusoidal positional encoding has been the go-to method for encoding positional information in computer vision ever since its introduction in [96]. For one-dimensional pixel values it can be defined as follows:

$$\begin{aligned} \mathrm{P(k, 2i)} &= \sin\left(\frac{\mathrm{k}}{\mathrm{n^{2i/d}}}\right) \\ \mathrm{P(k, 2i+1)} &= \cos\left(\frac{\mathrm{k}}{\mathrm{n^{2i/d}}}\right) \end{aligned} \tag{3-5}$$

where k are the pixel values, d is a user-specified embedding dimension, and n is temperature, which is typically set to 10000. For 2D coordinates, the embedding space is typically doubled, and the sinusoidal positional encoding is applied to both the x and y coordinates. In our case of 3D coordinates (e.g. [97]), the embedding space is tripled and x, y and z coordinates are used in place of k. Because of the different range between pixel values and 3D coordinates, we empirically found that n should be set to 10 for aerial and 100 for ground coordinates. We used 36 as the embedding dimension for both aerial and ground coordinates. An example of the positional embedding dimension on the longitudinal coordinate of the aerial points is shown on Figure 3-9.

Other methods of positional encoding have been considered, which are outlined below. Some were significantly and some were marginally worse than the sinusoidal positional encoding.

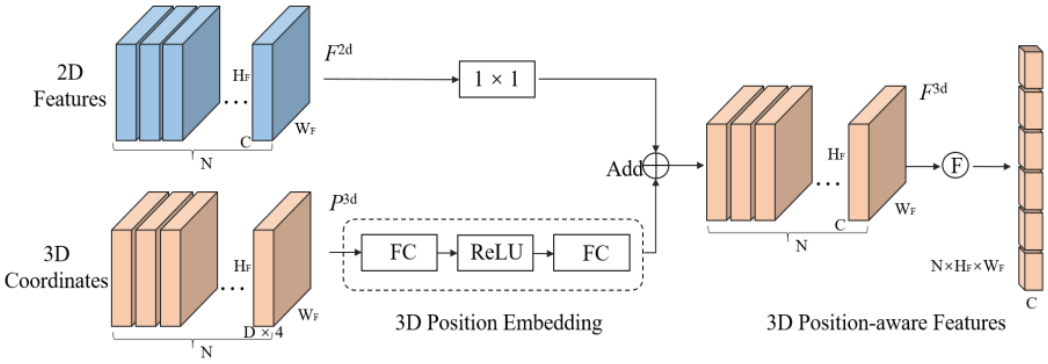


Figure 3-10: MLP embedding from PETR [80].

- **Simple position concatenation**. The simplest way is to simply concatenate the 3D coordinates to the feature vector. In the case of ground points, we would not concatenate 3D position, but rather the u-v coordinates and depth to ensure that position information is not used from the same (world) reference frame. However, we expect it to be difficult for a model to learn to optimize parameters associated with these coordinates - there are too few of them, despite being important.

- **Learned positional encoding**. Another option introduced in [96] is to learn the d-dimensional positional encoding for each pixel. Every time the token is used, a loss is applied on the positional embedding as well. Despite performing well in [96], there are two issues with it. The first is that [96] had to learn this embedding for $16x16$ patches, which is a lot fewer than the number of points we have, which can be two orders of magnitude larger. The second is that there is little theoretical justification for why this should outperform other methods in our case. It is hard to justify having to re-learn the positional information for each point, even though it is available. In [96] one of the benefits of learning this embedding is that it could capture dependencies beyond the Euclidean distance, e.g. that a patch two units to the right is more similar to a query patch than a patch one unit up and one unit to the right, despite being farther. This is because of the prior that objects typically stretch horizontally and vertically, but not diagonally. Such a prior would be extremely difficult to learn with 3D points anyway.

- **Positional Embedding through MLP**. The last option, particularly used in a similar setting to ours, is to use a multi-layer perceptron (MLP) to learn the positional embedding from the 3D position. Please refer to Figure 3-10 for more details.

Despite the sinusoidal embedding being clearly superior, which variables to embed remains an open problem. We hypothesize **AH3** that the best positional embedding method is the one that embeds the most information, i.e. one that embeds u, v and d, where u and v are pixel coordinates and d is the depth of a point. The intuition is that in the ground view, for instance, (u, v) coordinates introduce global information to local features. For instance, if a line is detected in the middle of a ground view, it is likely to be a road marking, while the same line at either corner is more likely to be a curb. On the other hand, excessive positional information might be detrimental to the locality of the features.

3-6-4 Tokenization

Once both the semantic and positional features at each point are known, they need to be combined together into a single descriptor.

The **aerial tokens** are constructed from the pruned aerial point set, $P_{a,p} \in \mathbb{R}^3$. For clarity we drop the subscript "p" (pruned) from now on. To construct the tokens, we first project the points onto the aerial image coordinates to obtain corresponding set of aerial image coordinates $C_a \in \mathbb{R}^2$. Afterwards, we use this coordinate set to look up the corresponding feature in F_a and obtain a set of features $T_{a,nopos}^{128\times N_a}$, where 128 is the number of channels in F_a. Afterwards, we concatenate positional information along the channel dimension to $T_{a,nopos}^{128\times N_a}$ to obtain the aerial token set $T_a^{S_a\times N_a}$, where S_a depends on the positional encoding method, described above. At the end, we apply a single layer MLP with a ReLU activation layer to incorporate positional information into the features. The MLP layer has S_a input channels and $\min S_a, S_g$ output channels.

In the same way, we construct **ground tokens** from the ground point set $P_g \in \mathbb{R}^3$ by first projecting them to the set of ground pixel coordinates $C_g \in \mathbb{R}^2$ to arrive at position-less ground token set $T_{g,nopos}^{128\times N_g}$. The $P_g \in \mathbb{R}^3 \rightarrow C_g \in \mathbb{R}^2$ mapping is based on true camera intrinsics and an identity matrix as camera extrinsics, thus assuming a camera centered at the world coordinate frame. We similarly use one of the positional encoding methods to concatenate positional information along the channel dimension. The position-embedded ground token set is $T_g^{S_g\times N_g}$, where S_g depends on the positional encoding method. If the same positional encoding method for both sets of points is used, $S_g = S_a$. At the end we apply a single layer MLP with a ReLU activation layer to incorporate positional information into the features. The MLP layer has S_g input channels and $\min S_a, S_g$ output channels.

3-6-5 Point Cross Attention

The next step is to use the aerial point tokens to predict the positions of ground tokens. This is done with a simple attention mechanism, where the ground tokens are the queries and the aerial tokens are the keys. The values are 3D coordinates of the aerial points in the world coordinate frame. The attention mechanism is defined as follows. First a linear projection is applied to the aerial tokens to arrive at $Q = T_g W_q$, where W_q is a learnable weight matrix. Then, the ground tokens are projected to arrive at $K = T_a W_k$. W_q and W_k are learnable weight matrices.

The attention scores are then calculated as follows:

$$\mathrm{A}' = QK^T \in \mathbb{R}^{N_g\times N_a} \tag{3-6}$$

We also propose a variant, where we do not attend to the aerial points that are too far away from the ground query. This is done by setting the attention score to $-\infty$ for the elements of A' which correspond to 0 entries in the mask matrix M. More formally, we performed a masked fill operation on A' with reference to the mask matrix M as follows:

$$a_{ij,f} = \begin{cases} a_{ij}, & \text{if } M_{ij} = 1. \\ -\infty, & \text{otherwise.} \end{cases} \quad \forall\ a_{ij} \in \mathrm{A}' \tag{3-7}$$

$$\mathrm{A}'_{\mathrm{f}} = \left[\{a_{ij,f}\}_{i=1}^{N_g} \times \{a_{ij,f}\}_{j=1}^{N_a} \right] \tag{3-8}$$

Afterwards, we can apply a softmax operation to the attention scores to arrive at the attention matrix A:

$$\mathrm{A} = \mathrm{softmax}\left(\frac{\mathrm{A}'_{\mathrm{f}}}{\sqrt{S_g}} \right) \tag{3-9}$$

Finally, we can use this attention matrix to predict the 3D coordinates of the ground points. This is done by multiplying the attention matrix with the values V:

$$\mathrm{P}_{\mathrm{pred}} = \mathrm{A}V \tag{3-10}$$

The values V are the 3D coordinates of the aerial points in the world coordinate frame. Note that no linear projection is applied to V as we expect a linear combination of the aerial points to be a good approximation of the ground point.

Furthermore, we predict the confidence of each ground point prediction by evaluating the weighted variance in positions of aerial points by the corresponding attention scores:

$$\sigma_i^2 = \frac{\sum_{j=1}^{N_a} a_{ij}(v_j - \mu_i)^2}{\sum_{j=1}^{N_a} a_{ij}} = \sum_{j=1}^{N_a} a_{ij}(v_j - \mu_i)^2 \tag{3-11}$$

3-6-6 Loss

After the locations for the ground points are predicted, we can calculate the loss. To do so, we first use the true camera extrinsics (x_e, y_e, ψ_e) to transform the 3D coordinates of the ground points (P_g) and obtain P_{true}.

There are two options to compute the loss. The first option is a simple MSE loss between the predicted and true 3D coordinates:

$$\mathrm{L}_{\mathrm{MSE}} = \frac{1}{N_g} \sum_{i=1}^{N_g} \left\| \mathrm{p}_{\mathrm{pred}_i} - \mathrm{p}_{\mathrm{true_i}} \right\|_2^2 \tag{3-12}$$

The second option is to optimize the negative log-likelihood of the distribution of the normalized attention scores A from Equation 3-9. Formally, for query point i, we calculate the mean and variance of the attention scores weighted by V using Equation 3-10 and Equation 3-11 respectively.

These means and variances are then used to calculate the negative log-likelihood of each ground truth 3D coordinate $p_{true} \in P_{true}$ under the distribution of the normalized attention scores:

$$\mathrm{L}_{\mathrm{NLL}} = \sum_{i=1}^{N_g} \left(0.5 \log(2\pi) + \log(\sigma_i^2) + \frac{(p_{true_i} - p_{pred_i})^2}{\sigma_i^2} \right) \tag{3-13}$$

We hypothesize **AH4** that negative log likelihood loss is better suited for local feature matching, since local features are expected to be unimodal. Thus, reducing the variance of attention scores around the ground truth is also a good proxy for reducing the localization error.

3-7 R2P Matching

The R2P matching architecture is largely similar to the P2P one described in the previous section, except in place of the ground point set $P_g = \{p_g^1, ..., p_g^{N_g}\}, p_g^i \in \mathbb{R}^3 \; \forall \; 1 \leq i \leq N_g$, we have a set of ground rays $R_g = \{r_g^1, ..., r_g^{N_g}\}, r_g^i \in \mathbb{R}^2 \; \forall \; 1 \leq i \leq N_g$.

The point/ray generation, pruning and feature extraction steps are identical to the P2P matching architecture. The difference in generating rays is that rays are generating uniformly in ground pixel coordinates, rather than 3D space, as shown on Figure 3-6a. The positional encoding and tokenization of the ground rays is also identical, except that depth embedding is not possible, so only the pixel coordinates are used. Furthermore, the cross-attention mechanism is also identical. Now instead of predicting where each ground point should be located, we predict where each ground ray should pass through.

The methodological difference in the two approaches lies in the loss function. In the P2P formulation the loss was calculated per point, based on the Euclidean distance between the predicted and true 3D coordinates. In the R2P formulation, the ground truth depth information is not available, so instead angular errors are used with either the MSE or the NLL loss, as defined in Equation 3-12 and Equation 3-13 respectively.

3-8 Camera Pose Estimation from Local Matches

The final step in the P2P and R2P matching architectures is to estimate the camera pose from the local matches. We opt to perform supervision on the local features and thus we perform this step outside of the training objective. There are two key ways to gen an estimate on the camera pose: the iterative pose refinement and the hypothesis formulation.

The camera pose estimation objective in the P2P formulation is to retrieve the camera pose $X_c = \{x_c, y_c, \psi_c, \theta_c, \phi_c\}$ subject to the following:

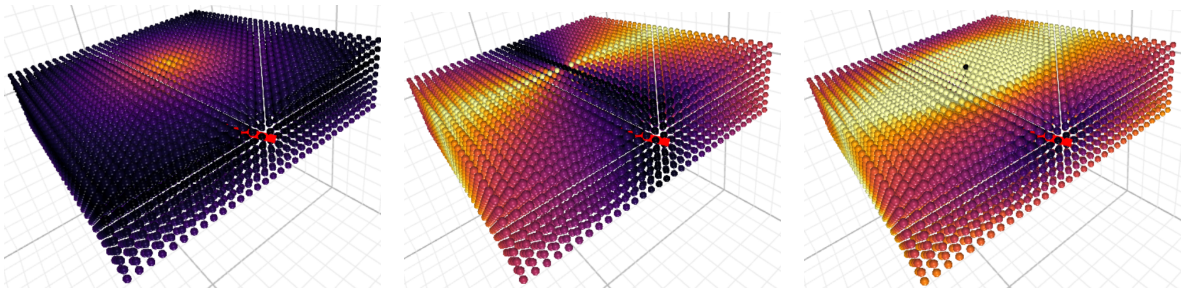
$$\hat{X}_c = \arg\min_{X_c} \sum_{i=1}^{N_g} \sigma_i^2 \left\| (R_{X_c} p_{g,i} + t_{X_c}) - p_{pred,i} \right\|_2^2, \tag{3-14}$$

where $p_{g,i}$ is the i-th ground point, $P_{pred,i}$ is the predicted location for the i-th ground point as calculated by Equation 3-10, R_{X_c} is the rotation matrix and t_{X_c} is the translation vector corresponding to the camera pose X_c. Finally, σ_i^2 is the predicted confidence as defined by Equation 3-11.

In R2P formulation, instead of dealing with Euclidean point differences, we evaluate angular differences between the ground ray and the angle that each predicted point makes with the camera center. The objective can thus be defined as follows:

$$\hat{X}_c = \arg\min_{X_c} \sum_{i=1}^{N_g} \sigma_i^2 \, \texttt{angnorm}_k(\lambda_{g,i} | r_{g,i}, X_c - \lambda_{pred,i} | p_{pred,i}, X_c), \tag{3-15}$$

where $r_{g,i}$ is the i-th ground ray, $p_{pred,i}$ is the predicted location for $r_{g,i}$. $\lambda_{g,i} \in \mathbb{R}^2$ is the angle that $r_{g,i}$ makes with the camera pose X_c. $\lambda_{pred,i} \in \mathbb{R}^2$ is the angle that $p_{pred,i}$ makes with the camera pose X_c. Finally, σ_i^2 is the predicted confidence as defined by Equation 3-11.



(a) Point errors due to pitch differences.

(b) Point errors due to yaw differences.

(c) Point errors due to pitch & yaw combined.

Figure 3-11: Aerial point errors with the query ray (red). Lighter color indicates higher error. These errors are weighted by the attention scores between the query ray and each aerial point and summed to arrive at the final error, given a hypothesis, which defines the origin of the query ray. In Figure c, the errors are combined via Equation 3-16 with $k = 3$.

Furthermore, $\texttt{angnorm}_k$ represents weighted normalization of the yaw difference with the pitch difference scaled by k, as shown on Equation 3-16. An example of the errors is visualized on Figure 3-11.

$$\texttt{angnorm}_k(\lambda) = \texttt{angnorm}_k\left(\begin{bmatrix}\Delta\theta\\ \Delta\phi\end{bmatrix}\right) = \sqrt{\Delta\theta^2 + \Delta\phi^2 k^2} \tag{3-16}$$

3-8-1 Hypothesis Formulation

In the hypothesis formulation X_C in equations above represents a discrete set of hypotheses. This allows to evaluate the function that is being minimized for all hypotheses in parallel and then select the camera pose that minimizes the function. However, of course, there is no guarantee that the global optimum will be found.

3-8-2 Iterative Pose Refinement

Both Equation 3-14 and Equation 3-15 are non-linear functions where the camera pose X_C depends on itself. Therefore, we also attempt to optimize the pose iteratively using a nonlinear least squares solver subject to constraints, based on the localization setting. We use the standard `scipy` library implementation of the solver.

3-8-3 Point Differences

Particularly in the P2P formulation, we can also use the point differences as a proxy for the camera pose. This is done by simply subtracting the predicted and true 3D coordinates and using the differences to predict the camera pose. While orientation estimation is not possible and lateral point errors are significantly affected by the camera orientation, point differences might still be somewhat informative about the longitudinal error.

Despite the numerous advantages of the hypothesis formulation, we hypothesize that iterative pose refinement is better suited for the task of fine-grained localization due to the absence of an irreducible error, which is present in the hypothesis formulation due to sampling density.

3-9 Implementation Details

In this section we provide the implementation details necessary to ensure reproducibility of the three methods. First, we describe the feature extraction backbone, common to all three approaches in subsection 3-9-1. Then, we identify the hyperparameters and design choices per approach in subsection 3-9-2, subsection 3-9-3 and subsection 3-9-4 respectively. Finally, we describe the hyperparameters and design choices for camera pose estimation from local features in subsection 3-9-5.

3-9-1 Feature Extraction

In all approaches we use a ResUNet-like [98] backbone to extract features either from the original images (I_a, I_g) or from one original and one projected image (in the case of the Regression approach). The architecture is shown on Figure 3-12. The architecture of a decoder block used to retrieve blue blocks is inspired by [35] and shown on Figure 3-13

3-9-2 Regression Approach Implementation Details

We train the model with a batch size of 8, learning rate of 0.001 and Adam optimizer [100]. The model is trained for up to 10 epochs depending on the remaining hyperparameters, until the performance on the cross-area set stops improving. The resulting feature maps from ResUNet are reduced to 6 output channels using an MLP with 5 linear layers, each halving the number of outputs, while the last brings it down from 256 to 6.

3-9-3 P2P Matching Approach Implementation Details

When spawning ground and aerial points, it was found that spawning uniformly in the 2D image domain and projecting each point at a number of depths yields best performance. The aerial points are spawned every 8 pixels both along the width and height axis of the aerial image. The 3D points are created by projecting the aerial points every 1.5 m along the height dimension. The ground points are spawned every 16 pixels along the width and height axis of the ground image. The 3D points are created by projecting the ground points every 2 m along the depth dimension.

Point pruning is performed to get rid of points that would not be a possible match, given any possible camera pose within the uncertainty region. For this we defined the edge cases for the maximum pitch, roll and altitude as 5°, 5° and 1 m. Additionally, the uncertainty region was expanded by a factor of 1.5 to include points that are not correct matches, but allow the model to predict the camera pose at the very edge of the uncertainty region.

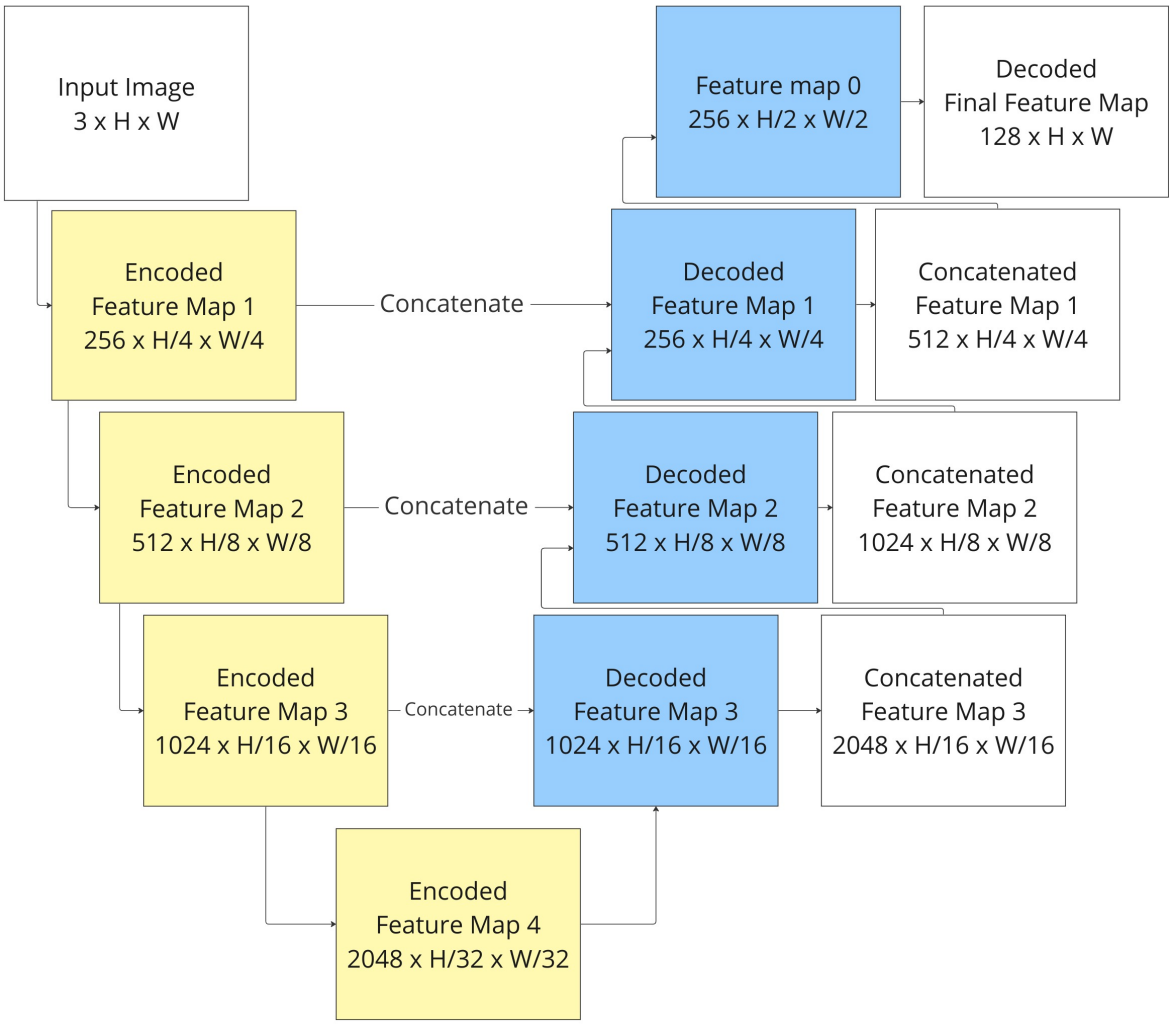


Figure 3-12: Backbone Architecture. Yellow encoder blocks are ResNet blocks, pre-trained on ImageNet [99]. Blue decoder blocks are created and trained from scratch, with architecture shown on Figure 3-13. Values represent dimensionality, where H and W are the height and width of the input image.

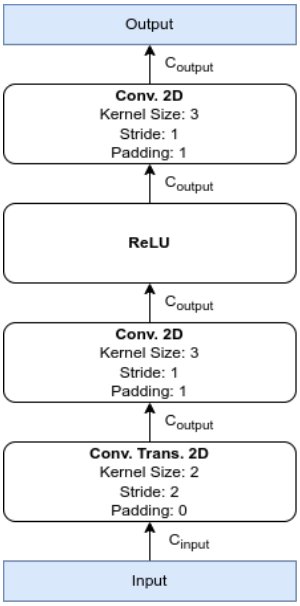


Figure 3-13: Decoder Block

We train the model with a batch size of 1 for up to 15 epochs, depending on the remaining hyperparameters, until the performance on the cross-area set stops improving.. The learning rate is set to 1×10^{-5} with ADAM optimizer [100].

3-9-4 R2P Matching Approach Implementation Details

In R2P matching we also spawn ground rays and aerial points uniformly in the image plane. Aerial points are projected to the 3D space at discrete intervals. For the ground image these are 32 pixels along the width and height dimension to form 256 ground rays. For the aerial image, these remain 16 pixels along the width and height dimension, but are projected every 3 m along the depth dimension. The same edge cases defined in subsection 3-9-5 are used for pruning.

We train the model with a batch size of 6 on the KITTI dataset and 7 on ORC due to the different sizes of input images. The learning rate is 1×10^{-4} with ADAM optimizer [100]. The model is trained for up to 10 epochs, depending on the remaining hyperparameters, until the performance on the cross-area set stops improving.

3-9-5 Pose Esimation from Local Features Implementation Details

To predict the pose with iterative refinement we run the bounded Trust Region Reflective (TRF) least squares solver for up to 600 iterations, since it outperformed Levenberg-Marquardt (LM) in our experiments. The bounds on predicted coordinates are determined by the localization setting and the edge cases defined above.

To predict the pose with a hypothesis formulation we use the number of hypothesis per dimension as shown on Table 3-2.

Table 3-2: Number of hypotheses along each dimension in the R2P localization algorithm.

	Longitudinal	Lateral	Altitude	Roll	Pitch	Yaw	**Total**
3-DoF	8	8	1	1	1	8	**512**
6-DoF	4	4	1	3	3	4	**576**

3-10 Hypotheses Overview

In order to guide the design of the experiments, we formulate **key hypotheses (KH)** that we will test via experiments. The key hypotheses are listed below.

1. **KH1**: Local feature matching (P2P and R2P matching) offer a viable, interpretable alternative to the baseline approach.
2. **KH2**: Due to occlusions and empty space, R2P matching is both more accurate and more interpretable than P2P matching.
3. **KH3**: Pitch and roll angles can be predicted accurately via local feature matching approaches.
4. **KH4**: Hypothesis formulation is better suited for camera pose estimation than iterative pose refinement due to the ability to capture the global optimum.

Furthermore, in this chapter we have formulated a number of hypotheses, which are specific to each method. These hypotheses will motivate the ablation study for each method, and thus are referred to as **ablation hypotheses** (**AH**). The method-specific, ablation hypotheses were defined in this chapter and are reiterated below.

Regression Approach

- **AH1**: Extracting features before projecting them is more effective than projecting the original images.
- **AH2**: Projecting the aerial image onto the ground domain is more effective than projecting the ground image onto the aerial domain, which is still more effective than not projecting at all.

Local Feature Matching Approach

5. **AH3**: The best positional embedding method is the one that embeds the most information, i.e. one that embeds u, v and d, where u and v are pixel coordinates and d is the depth of a 3D point.
6. **AH4**: Negative log likelihood loss is better suited for local feature matching, since local features are expected to be unimodal. Thus, reducing the variance of attention scores around the ground truth is also a good proxy for reducing the localization error.

Chapter 4

Experiments

Having outlined the methodologies and a set of hypotheses in chapter 3, we now attempt to answer the research questions defined in section 1-4 and hypotheses defined in section 3-10 through a set of experiments. The section overview is presented in Table 4-1.

We define three experimental settings, as presented on Table 4-2. The "wide" setting is used primarily for clarity of visualizations; "coarse" setting is used to compare with traditional localization methods, which are typically evaluated on this setting. Finally, the "fine" setting is used to evaluate the performance on the proposed novel, "fine" localization setting. Results are on KITTI dataset, unless otherwise specified.

More specifically, we start off by discussing the quantitative performance of the three proposed methods on the three localization settings in section 4-1. There we also compare the methods with the existing baseline and comment on the time complexity. In section 4-2 we discuss the qualitative performance of the proposed local feature matching methods, specifically regarding the capacity to be interpretable. In section 4-3 we compare the three methods to retrieve the camera pose from the local features based on accuracy and speed. Furthermore, in section 4-4 we explore how well the pitch and roll angles are be predicted in the 6-Degree of Freedom (DoF) setting.

In section 4-5 we discuss the limitations of the proposed local feature matching methods, where we conclude that "unmatchable" points/ray pose the largest issue together with the lack of a sufficient number of features that are discriminative laterally, given forward-facing cameras. In section 4-6 we explore the key issue with the Point-to-Point (P2P) matching algorithm and justify why Ray-to-Point (R2P) is our largest contribution, despite worse performance. Finally, in section 4-7 we perform an ablation study to answer the ablation hypotheses identified in chapter 3. These are less interesting design choices, which are not directly related to the research questions, but are still important to answer.

Table 4-1: Overview of hypotheses and corresponding sections.

Hypothesis Label	Hypothesis Description	Section(s)
KH1	Local feature matching (P2P and R2P matching) offer a viable, interpretable alternative to the baseline approach.	section 4-1, section 4-2, section 4-5
KH2	Due to occlusions and empty space, R2P matching is both more accurate and more interpretable than P2P matching.	section 4-1, section 4-6
KH3	Pitch and roll angles can be predicted accurately via local feature matching approaches.	section 4-4
KH4	Hypothesis formulation is better suited for camera pose estimation than iterative pose refinement due to the ability to capture the global optimum.	section 4-3

Table 4-2: Experimental settings.

Setting Name	x_{max}	y_{max}	ψ_{max}
"Wide"	20 m	20 m	360°
"Coarse"	20 m	20 m	10°
"Fine"	5 m	5 m	10°

4-1 Quantitative Comparison Between Methods and with Baselines

In this section we compare the performance of the three proposed methods between each other and with the state-of-the-art baseline. The results for the "fine" localization setting are presented in Table 4-3, results for the "coarse" setting and presented in Table 4-4 and for "wide" in Table 4-5.

*The values are reported by follow-up work [94]. These values are median errors, not mean errors. Additionally, the setting has 15° orientation uncertainty, not 10°. Finally, the cross-area results are reported for the FordAV dataset, when trained on KITTI, not test2 split of KITTI.

Table 4-3: Mean errors on the "fine" setting, as defined by Table 4-2. Best performing model on same-area setting is selected. Results are on the KITTI dataset, unless otherwise stated. HA abbreviates HighlyAccurate [33] and is the baseline in the "fine" localization setting. Best result is highlighted in bold.

	Train			Same-Area			Cross-Area		
	Long. (m)	Lat. (m)	Orient. (deg)	Long. (m)	Lat. (m)	Orient. (deg)	Long. (m)	Lat. (m)	Orient. (deg)
HA reported*	1.97	0.63	1.40	2.01	0.83	1.82	3.11	3.17	6.59
HA reproduced	1.88	0.57	1.02	2.09	0.62	1.24	2.74	2.18	2.08
Regression	**0.46**	**0.18**	**0.37**	**0.68**	**0.25**	**0.40**	**2.41**	**1.20**	1.75
P2P	1.97	0.29	0.41	2.05	0.34	0.48	2.68	1.81	**1.43**
R2P	2.71	1.3	1.52	2.72	1.36	1.5	3.24	2.20	2.18

Table 4-4: Performance of the proposed R2P method and the current state-of-the-art baselines on the "coarse" setting of the KITTI dataset, as defined in Table 4-2.

	Same-Area				Cross-Area			
	Localization (m)		Orientation (deg)		Localization (m)		Orientation (deg)	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
HighlyAccurate	12.08	11.42	3.72	2.83	12.58	12.11	3.95	3.03
SliceMatch	7.96	4.39	4.12	3.65	13.50	9.77	4.20	6.61
CCVPE	**1.22**	**0.62**	**0.67**	**0.54**	**9.16**	**3.33**	**1.55**	**0.84**
R2P	9.40	7.14	4.38	3.53	13.85	11.34	5.27	4.28

Table 4-5: Performance of the proposed R2P method and the current state-of-the-art baselines on the "wide" setting of the KITTI dataset, as defined in Table 4-2.

	Same-Area				Cross-Area			
	Localization (m)		Orientation (deg)		Localization (m)		Orientation (deg)	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
HighlyAccurate	15.51	15.97	89.91	90.75	15.50	16.02	89.84	89.85
SliceMatch	9.39	5.41	**8.71**	**4.42**	14.85	11.85	**23.64**	**7.96**
CCVPE	**6.88**	**3.47**	15.01	6.12	**13.94**	**10.98**	77.84	63.84
R2P	17.74	14.31	48.39	57.68	18.86	15.90	51.14	61.25

Quantitative results on the "fine" setting"

Table 4-3 shows the results for the "fine" setting, which is the only one which compares all three of the proposed models. Clearly the regression approach outperforms all other approaches, including the HighlyAccurate [33] baseline.

There are two significant differences between the regression approach and HighlyAccurate: the lack of iterative optimization and a more modern ResUNet backbone, which is pre-trained on ImageNet. Levenberg-Marquardt (LM) optimization offers questionable performance improvement, as shown in Appendix B. The improved backbone, especially when pre-trained is likely also a large contributor to the improved performance.

When compared with P2P and R2P, the regression approach also demonstrates the highest degree of overfitting. This is not surprising, given that it is the only approach of those three whose loss is applied on the final pose, rather than intermediate local features. Overall, neither of the approaches demonstrated performance, which was even remotely consistent with the localization requirements described in subsection 2-1-3.

Both the regression approach and P2P outperform R2P significantly, especially in terms of orientation performance. Neither P2P nor R2P were subjected to any significant hyperparameter tuning, thus comparison with state-of-the-art is not entirely fair. Therefore, in this thesis we focus on the R2P approach, which, despite worse performance, is more interpretable than the regression approach and more robust to the road prior than P2P. The latter argument is especially vital for generalization to unseen environments, as it ensures that truly local features are being matched. These conclusions are drawn in section 4-2.

Table 4-6: Speeds of the proposed methods compared with the HighlyAccurate baseline.

	HighlyAccurate from [33]	Regression	P2P	R2P
Mean Speed	500	33	178	106
Median Speed	-	33	178	106

Quantitative results on the "coarse" and "wide" settings"

For this reason in Table 4-4 and Table 4-5 we only compare the R2P approach with the existing state-of-the-art. On the "coarse" localization setting, the proposed R2P approach performs similarly to SliceMatch [34], especially in terms of orientation error, while being slightly worse in terms of translation error. CCVPE [35], however, still outperforms all other approaches by a significant margin.

On the "wide" localization setting, the proposed R2P approach performs poorly. The proposed approach was not tailored for the wide localization setting, where distant points may be the correct match. Moreover, HighlyAccurate [33] performs almost equally as poorly and completely fails to capture orientation. CCVPE [35] also fails to capture orientation on the cross-area test set. Moreover, a large component of relative success of SliceMatch [34] is the 64 hypothesis bins that the authors use to estimate orientation, higher than both R2P and CCVPE. However, such a localization setting is not reasonable for autonomous driving, as "fine" localization setting was identified as the industry target.

Time complexity of the proposed methods

The speeds of the approaches are presented on Table 4-6. In subsection 2-1-3 we established that for online operations, the localization algorithm should operate at least at 5 Hz. All proposed approaches meet this requirement.

Compared with other approaches on the "coarse" setting, the P2P and R2P methods a only slightly slower. The Regression approach, however, is only outperformed by SliceMatch [34] in terms of inference speed.

4-2 Interpreting Local Feature Matching Methods

In subsection 2-3-7 we discussed the lack of interpretability of most of the state-of-the-art methods. We also identified two levels of interpretability: one on the camera pose level and one on the local feature level. In this section we demonstrate that the proposed local feature matching methods (i.e. P2P and R2P) are interpretable on both levels. We will use the R2P method as an example.

4-2-1 Interpreting on the Local Feature Level

The architecture of P2P and R2P algorithms allows to visualise the prediction for each query ray/point. One example is shown on Figure 4-1. To get more insight into the model's

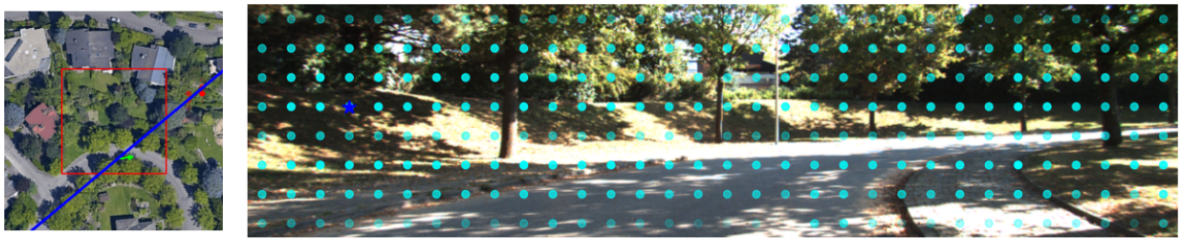


Figure 4-1: Example of the R2P algorithm predicting the location of a query ray on the test1 split of the KITTI dataset. **Ground image:** dark blue star - query ray, cyan dots are all possible queries, opacity of each cyan dot is proportional to the ground truth error of the corresponding query ray. **Aerial image:** red dots - aerial points, which match highly with the query ground ray. Opacity of each red dot is proportional to the attention score. Green arrow - ground truth vehicle location. Blue line - ground truth query ray. Red square - uncertainty on the vehicle location.

Table 4-7: Links to the videos of the ray sweep for the R2P algorithm on the KITTI dataset.

Example	Dataset	Sample	Mean Angle Error (deg)	Link
1	Train	Best	3.89	https://youtu.be/aXJRCOFoTNO
2	Test1	Best	4.05	https://youtu.be/qaMNpXHFdzc
3	Test2	Best	4.61	https://youtu.be/An6x5O64syc
4	Train	Worst	41.1	https://youtu.be/uczsq6aXrms
5	Test1	Worst	16.4	https://youtu.be/hs9xq5O-xkO
6	Test2	Worst	56.3	https://youtu.be/TVYUW955vOg

behaviour, however, the ray sweep can be visualized in a video format. The links are provided in Table 4-7.

Note that in general, the aerial points are matched well with the query ray. The mean angle errors are typically between 5° and 10°. Sometimes the high errors are difficult to explain, such as in examples 4 and 6, but such examples are rare.

An important observation, however, is the fact that sometimes the predictions are "too accurate", as is the case on Figure 4-1. The predicted location of the query ray is good, but it would be impossible to predict it using semantics alone, given that the query ray is clearly occluded earlier than where the aerial matches are found.

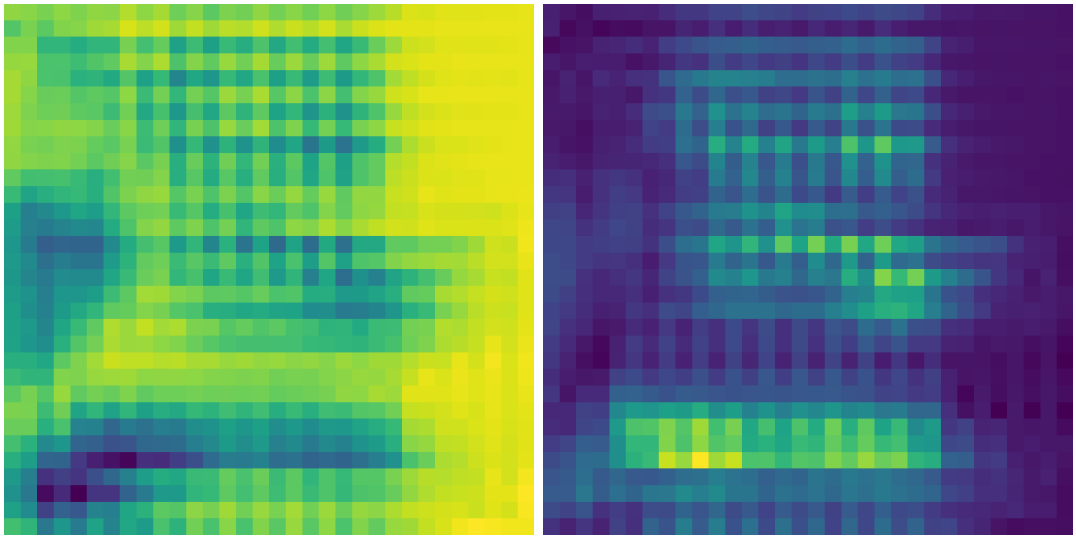
There are two possible explanations for this. First is that the model is exploiting the road prior, which is the main issue with the P2P algorithm, specifically demonstrated on Figure 4-9b. Using the road prior, the model might be predicting the location of the vehicle and offsetting all aerial matches based on the vehicle location and the implicit representation of the ground ray coordinates.

This time, however, such an explanation is less likely. First of all, the model has some depth perception, as shown on example 5 on Table 4-7. The aerial matches make a jump in distance from the camera at the 1 s timestamp, when the ground view semantics switch from background, which is beyond the aerial image scope, to a building, which is present on the aerial image. Such a jump in distance is only possible if semantics are matched between the two views, which means that the road prior is at least not used exclusively.

Another reason is that this time, when the ground image is blacked-out, the performance worsens significantly, as opposed to the P2P algorithm. The comparison between the original

Table 4-8: Links to the videos of the ray sweep for the R2P algorithm on the KITTI dataset, for the original and blacked-out ground image at test time only.

Original Ray Sweep	Blacked-Out Ground Image Ray Sweep
https://youtu.be/-qUf67zosKQ	https://youtu.be/ZzDHCG-74tM



(a) Channel 0 **(b)** Channel 1

Figure 4-2: Example feature maps of the aerial image depicted on Figure 4-1. There are a total of 64 channels.

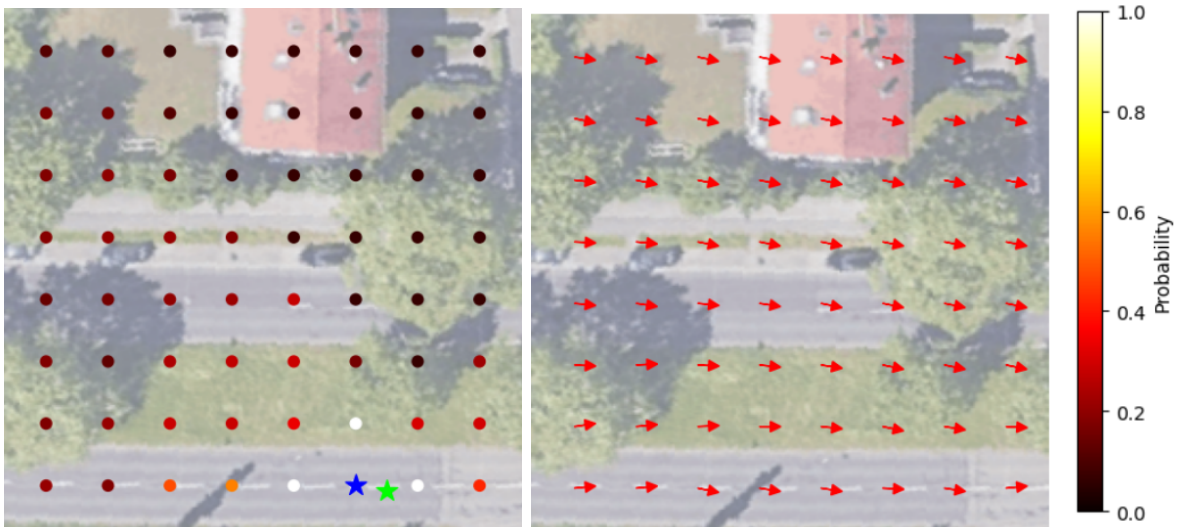
ray sweep and one with a blacked-out ground image at test time is given by Table 4-8. This was not the case for the P2P algorithm, where the performance was almost unaffected by the blacked-out ground image, as shown on Figure 4-9c.

Given that the model does not exploit the road prior exclusively, and yet the seemingly impossible matches are accurate, the second explanation is that semantics are being used implicitly to make a prediction that minimizes the angle, yet does not necessarily correspond to the aerial match at the correct depth. As described in section 3-7, the loss function is applied to the angular error of each ray. Thus, there is nothing ensuring that the point with correct semantics will be predicted. Due to the deep backbone, the model can learn to propagate semantic features along each ray, thus resulting in a feature map that is not geometrically representative of the original image - i.e. features that are stretched along the ray direction.

Such a conclusion, however, cannot be proved using the feature maps alone, e.g. Figure 4-2. Clearly the feature maps contain the road structure not the ray-like structure anticipated. However, the combination of 64 channels may be inducing the ray dispersion of feature, which is not possible to observe without a PCA-like analysis.

4-2-2 Interpreting on the Camera Pose Level

On the camera pose level, probability heatmaps can be used to visualize the model's confidence in the predicted camera pose and the direction of uncertainty, similar to [34, 35, 39].



(a) Position heatmap **(b)** Best orientation bin **(c)**

Figure 4-3: Probability heatmaps for the R2P algorithm on the KITTI dataset. The heatmaps are generated using 8 hypotheses for each dimension in 3-DoF. Blue star - ground truth camera location. Green star - predicted hypothesis. (c) Colormap used for the heatmaps throughout the thesis.

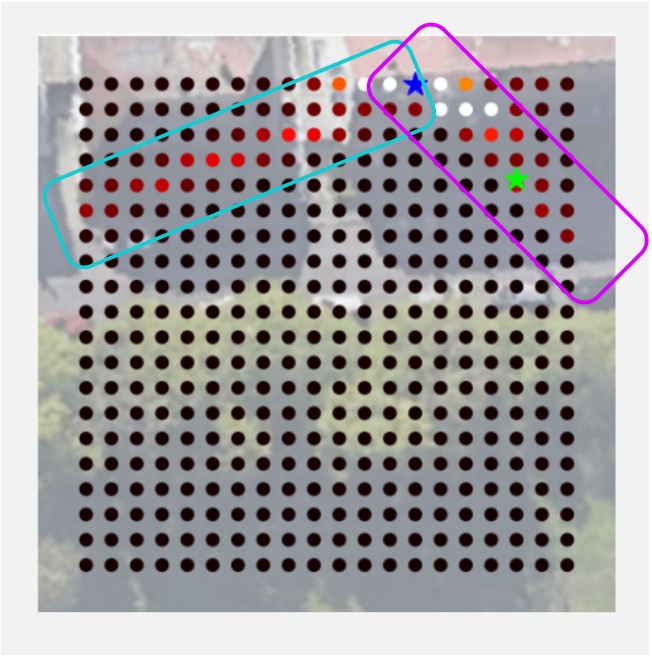
One example of such heatmaps is shown on Figure 4-3. Clearly, the predictions lay close to the ground truth and along the road. However, such a plot is not as informative, since the orientation bins are too close together to be able to distinguish between them. Unfortunately, as mentioned in section 4-1, we attempted to train the model on the "wide" setting, which would aid in interpreting orientation bins, but the resulting model's performance was significantly worse in relative terms compared to the "coarse" setting.

Therefore, for visualization purposes we disable the orientation dimension and increase the number of spawned hypotheses in longitudinal and lateral dimensions to 20. An example heatmap and the corresponding ground image are given on Figure 4-4. Two conclusions can be drawn from this figure.

First of all, note the high confidence in the pose prediction along the two rays, labelled in cyan and purple. This is a result of a high confidence in the ray that goes through those points on the aerial map, and approximately the corresponding labeled regions in the ground image. High confidence entails that at the ray matching module all aerial points with high attention scores with the query ground ray are very close together. This means that the Gaussian is very narrow if Gaussian weighting is used, or the mean error is small, if the mean error is used. In either case, the result is that all hypotheses along this ray are highly likely, but it is impossible to distinguish the location along the ray. On Figure 4-4 there are clearly two such rays and thus the location at the interception of the two rays is the most likely, as is the case. This proves that the model behaves as expected.

Secondly, note that the predicted location is on top of the building in Figure 4-4. It is difficult for the current formulation to exploit the road prior, as already established in subsection 4-2-1. The other two proposed methods, Regression and P2P do not have this property to such an extent. This is a positive result, since it means that quantitative performance can be

(a) Ground image



(b) Aerial image

Figure 4-4: Interpreting the heatmaps. Example from test1 on KITTI dataset.

Table 4-9: Comparison between camera pose estimation methods, given a set of point matches. Point Diff, Least Squares and Hypotheses are the three methods proposed in chapter 3. The best result for each metric is highlighted in bold. Mask refers to masking out matches that would be unfeasible, given any possible camera pose within the uncertainty specifications both at train and test time. Values represent mean absolute error.
*Altitude does not include ego-altitude on the KITTI dataset since the ground truth is not available.

			Longitudinal (m)	Lateral (m)	Altitude* (m)	Roll (deg)	Pitch (deg)	Orientation (deg)
Train	No Mask	Point Diff.	**1.53**	3.21	0.42	-	-	-
		Least Squares	1.97	**0.29**	**0.07**	**0.10**	**0.16**	**0.41**
		Hypotheses	1.63	0.60	-	1.01	1.21	0.78
	Mask	Point Diff.	**1.41**	3.23	0.45	-	-	-
		Least Squares	2.19	**0.26**	**0.11**	**0.20**	**0.25**	**0.49**
		Hypotheses	1.78	0.52	-	1.18	1.42	0.81
Test 1	No Mask	Point Diff.	**1.57**	3.30	0.62	**-**	-	-
		Least Squares	2.05	**0.34**	**0.08**	**0.30**	**0.32**	**0.48**
		Hypotheses	1.84	0.92	-	1.34	1.30	0.73
	Mask	Point Diff.	**1.41**	3.25	0.57	-	-	-
		Least Squares	2.21	**0.30**	**0.12**	0.50	**0.47**	**0.56**
		Hypotheses	1.80	0.78	-	1.38	1.45	0.83
Test 2	No Mask	Point Diff.	**2.86**	4.11	0.43	-	-	-
		Least Squares	3.00	1.97	**0.18**	0.66	0.77	1.41
		Hypotheses	2.92	**1.78**	-	1.49	1.60	**1.32**
	Mask	Point Diff.	**1.97**	3.58	0.43	-	-	-
		Least Squares	2.68	1.81	**0.11**	0.74	**0.38**	1.43
		Hypotheses	2.42	**1.68**	-	1.49	1.43	**1.40**

improved by weighing hypothesis by some prior likelihood based on the aerial image alone, as is done, for instance, in [35], if raw performance is desired. However, as discussed in chapter 2, incorporating the road prior is usually the opposite of what is desired in industrial settings. In any case, the ability of R2P to decouple the road prior from local feature matching is, to the best of our knowledge, a novelty in the field of visual localization.

4-3 Camera Pose Estimation from Local Features

In this section we compare the performance of the three proposed methods for camera pose estimation, discussed in section 3-8. The results are presented in Table 4-14. Note that the results are only reported for the P2P algorithm. This is because iterative pose refinement is not feasible for R2P matching, since it was found that computing the angles of each aerial point with each ray, given an iteration update in camera pose was too computationally expensive. In the hypothesis formulation, all angles are pre-computed for each hypothesis at the beginning of training, which takes up to 1 min for ~ 600 hypotheses. This is not feasible in real time, when ~ 500 iterations are required for convergence.

4-3-1 Localization Accuracy

Regarding the accuracy of the P2P algorithm, the most important observation is that point difference is consistently best in terms of longitudinal performance, while least squares is most often best in terms of other degrees of freedom. Point differences are also high in the lateral direction. This is expected. Minimizing point differences is the training objective, thus optimizing point longitudinal error is as important as lateral and as important as altitude. When optimizing the camera pose using least squares, however, lateral performance is coupled with orientation. Finding a camera pose that aligns all points perfectly in the lateral direction will result in a perfect lateral camera pose and a near-perfect orientation prediction. Finding a camera pose that aligns the points perfectly longitudinally almost exclusively contributes to longitudinal performance. Assuming there is a trade-off between aligning points laterally and longitudinally, it makes sense that the model is more conservative in the lateral direction, since minimising lateral error is more important for the 3/6DoF camera pose.

Another vital observation is in the difference between the hypothesis formulation and least squares. Hypothesis formulation performs consistently worse, especially so in terms of pitch and roll angles. Because of memory requirements, it is not possible to evaluate sufficiently many hypotheses at inference, which is the source of large hypotheses errors. The hypothesis formulation performance was found using 6 hypotheses longitudinally and laterally, 8 hypotheses in the orientation direction and 3 hypotheses for pitch and roll angles. This is significantly fewer than was used in SliceMatch, where at test time the authors use 15 hypotheses for longitudinal and lateral position each with 64 orientations.

Another observation is that as expected, the masked models perform better than the unmasked models. This is due to the fact that the mask helps get rid of impossible matches, which are a source of uncertainty for the model.

4-3-2 Localization Speed

The inference speeds for the local feature matching methods are given in Table 4-10. Note that iterative refinement is completely unfeasible in real time. The reason is that in the hypothesis formulation the distances or angles between each aerial point and the camera pose can be pre-computed. In iterative refinement, there is no way to pre-compute these distance, since the camera pose is being updated differently depending on the sample. Despite the fact that the numbed of tested camera poses is similar (iterative refinement converges in ~ 600 iterations, which is similar to the tested number of hypotheses on Table 3-2), iterative refinement is ~ 15 times slower.

Note that computing angles with aerial points per hypothesis is significantly slower than distances to them. Pre-computing can take up to 2 min for the hypothesis formulation, which makes it completely unfeasible to do so online, across 600 iterations. Therefore, R2P matching with iterative refinement was not tested.

4-4 Predicting Pitch and Roll Angles

In this section we attempt to address KH3, namely that pitch and roll angles can be predicted accurately via local feature matching approaches. The results of the R2P algorithm are

Table 4-10: Time complexity of proposed local feature matching methods. Statistics recorded over 100 iterations, after 3 warm-up iterations. Values are provided in ms.

		Stage 1: Determine Local Feature Matches	Stage 2: Determine Camera Pose via Hypothesis Formulation or Iterative Pose Refinement		Total Pose Estimation Time	
		Point/Ray Attention	Iterative Ref.	Hypothesis Form.	Iterative Ref.	Hypothesis Form.
Mean	P2P	86	2320	92	2410	178
	R2P	34	-	72	-	106
Median	P2P	85	1990	93	2070	178
	R2P	35	-	71	-	106

Table 4-11: Effect of including additional degrees of freedom in the R2P localization algorithm. The best result for each metric is highlighted in bold. Hypothesis formulation was used to determine camera pose. Values represent mean absolute error. For clarity, the best result is not highlighted in bold, but note that the 3-DoF model is consistently best for the three degrees of freedom that it predicts.

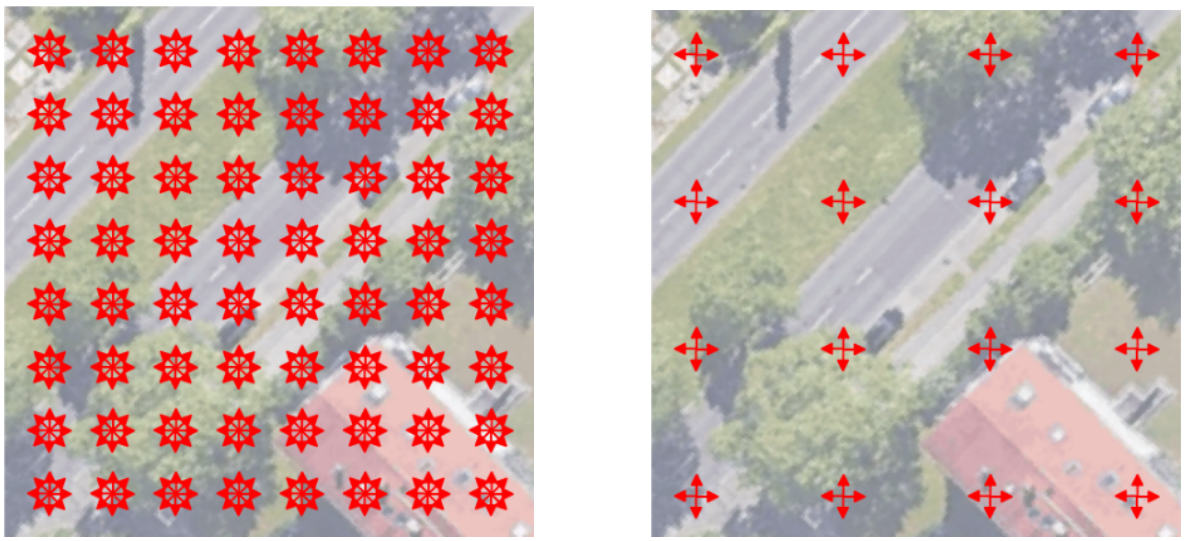
			Longitudinal (m)	**Lateral (m)**	**Roll (deg)**	**Pitch (deg)**	**Orientation (deg)**
Train	"Fine-grained"	3-DoF	2.71	1.3	-	-	1.52
		6-DoF	3.17	2.23	1.24	1.01	2.69
	"Coarse"	3-DoF	8.49	3.22	-	-	4.25
		6-DoF	9.88	5.22	1.56	1.68	7.15
Test1	"Fine-grained"	3-DoF	2.72	1.36	-	-	1.5
		6-DoF	3.09	2.23	1.19	0.94	2.69
	"Coarse"	3-DoF	8.8	3.31	-	-	4.38
		6-DoF	9.95	5.29	1.52	1.6	7.19
Test2	"Fine-grained"	3-DoF	3.24	2.2	-	-	2.18
		6-DoF	3.52	2.78	1.39	0.76	2.83
	"Coarse"	3-DoF	12.29	6.39	-	-	5.27
		6-DoF	12.91	7.77	1.49	1.41	6.87

presented on Table 4-11.

Note that on both the "fine-grained" and the "coarse" localization setting the performance on longitudinal, lateral and orientation coordinates is significantly worse when the model additionally predicts the pitch and roll angles.

The reason lays in the memory requirements. The R2P model, as described in section 3-7, has the loss applied on a ray level, rather than the final camera pose. The pose is therefore not determined end-to-end. After performing a forward pass, there is capacity to test approximately 600 hypotheses. When splitting between 3 degrees of freedom, we are able to dedicate more hypotheses towards these dimensions, as shown on Table 3-2, thus improving performance.

Instead of sampling the locations of pitch and roll hypotheses uniformly between $-\psi_{max}$ and ψ_{max} and $-\theta_{max}$ and θ_{max} respectively, we made use of the train dataset statistics to minimize the distance between each pitch and roll angle and its closest hypothesis. The resulting location of hypotheses and the distribution of the ground truth angles are shown on Figure 4-6.



(a) 3-DoF **(b)** 6-DoF

Figure 4-5: Example location of camera pose hypotheses for 3-DoF and 6-DoF localization, based on Table 3-2, in the top-down view. Clearly, the 6-DoF has too few hypotheses, which results in worse performance.

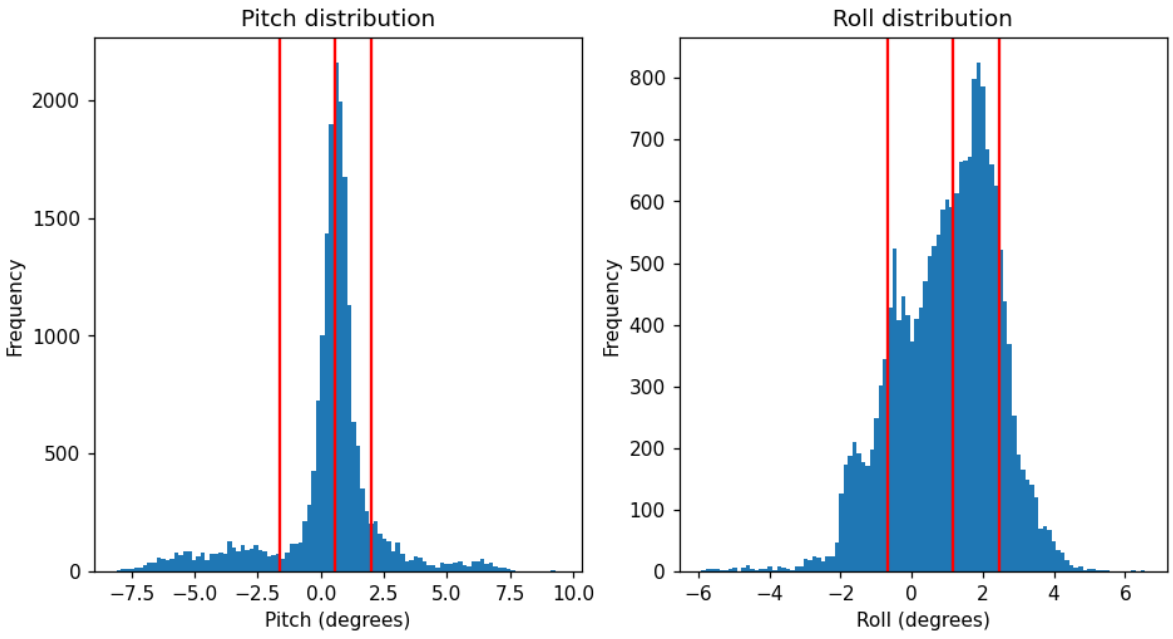


Figure 4-6: Location of pitch and roll hypotheses (red lines) and the distribution of the ground truth angles on the training dataset (blue).

Given this distribution of ground truths, the mean distance to the closest hypothesis, i.e. the irreducible error, is 76.7 cm for pitch and 47.3 cm for roll. The mean absolute error if zero pitch and roll angles are predicted would be 1.43 m for pitch and 1.47 m for roll on the train set. If random guesses were made instead, the errors would be many fold higher.

Therefore, the model is able to predict the pitch and roll angles beyond random guessing, but not far from the zero baseline. Testing with more hypothesis would likely improve the performance slightly. On the other hand, the 2D location prediction needs to be close enough to the ground truth for the pitch and roll predictions to be reasonable. For instance, in the "fine-grained" and "coarse" settings, the roll and pitch errors have the same distribution. But since the 2D location error is larger in the "coarse" setting, the pitch and roll errors are also larger.

The P2P algorithm, however, demonstrates significantly better pitch and roll performance, when least squares optimization is used to refined the camera pose, as shown on Table 4-14. This is a result of accurate point predictions in the altitude direction, which is the principal axis for predicting pitch and roll angles. The loss function, which treats longitudinal, lateral and altitude errors equally is more suited for accurate altitude predictions.

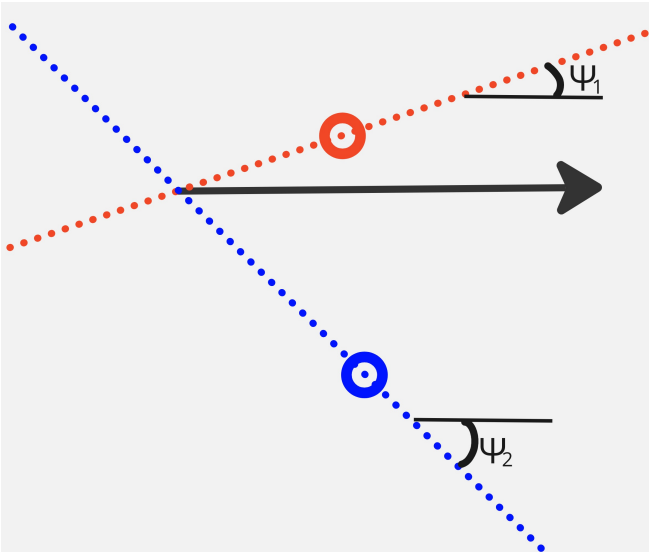
4-5 Limitations of Local Feature Matching Methods

Note that in general the aerial matches are quite close to the ground truth ray on all videos. However, this does not translate into sufficient quantitative performance, especially in the longitudinal direction. The lack of the road prior was already outlined as a significant contributor to the poor quantitative performance. However, there are two other factors that could play a role.

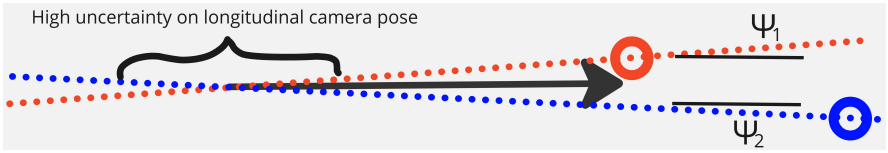
Firstly, the lack of explicit depth prediction for the R2P method was identified as a significant advantage in section 3-7. The reason was that accurate depth prediction within the required accuracy is oftentimes simply not feasible from the available semantic information, especially in the "fine" setting. However, this means that the longitudinal pose estimation is performed using points that are located to the side of the camera, as explained on Figure 4-7, which there are many fewer of. Nevertheless, this means that the R2P can be expected to perform significantly better when 360° FoV is available, whether through multiple perspective cameras or a panoramic image.

Secondly, because local features are matched, sometimes there is a lack of connection between the subjective difficultly of a query image and a high quality prediction. For instance, on Figure 4-4, the query ground does not seem simple to localize due to the lack of longitudinal cues. The local features from regions 1 and 2 also do not seem to be too discriminative, and yet the prediction for the ray from region 2 is correct. This is different from existing methods, e.g. CCVPE [35], where a discriminative road structure largely affects the quality of the prediction.

Thirdly, despite the fact that matching rays rather than points creates fewer queries, a lot of them are still not "matchable". This is because some rays either represent a uniform structure, such as a wall, or the aerial match lies beyond the boundaries of the aerial patch. Forcing the model to make an accurate prediction for all rays creates the problems similar to what was described in section 4-6. In fact, for a human only a few rays would suffice to



(a) Rays point to the sides of the vehicle.



(b) Rays point largely in the longitudinal direction.

Figure 4-7: Longitudinal pose estimation is performed using points that are located to the side of the camera. Suppose the ray matching algorithm determined that the red ray, which makes angle ψ_1 with the longitudinal axis of the vehicle must go through the aerial point depicted in red. The same is true for the blue ray and the blue aerial point. Suppose these are the only two rays with high confidence, thus driving the pose estimation. If ψ_1 and ψ_2 are small, the longitudinal pose estimation is more uncertain. In fact, when $\psi_1 = \psi_2 = 0$, the longitudinal pose estimation is impossible and when $\psi_1 = \psi_2 = \pi/2$, the lateral pose estimation is not possible. Due to using a single perspective camera, most rays point largely towards the front of the vehicle, which makes longitudinal pose estimation more uncertain.

deduce the camera pose, as shown on Figure 4-7a. Only a few rays can also be expected to be discriminative enough to contribute to a high-precision prediction, especially in the "fine" setting.

Determining which rays are discriminative is similar to determining which regions in an image contain objects of interest in the object detection task. There are many types of object detectors, a lot of which map queries from an Multilayer Perceptron (MLP), e.g. [101, 102]. This would be one way to determine the query ground rays. An alternative is to use learnable queries, which are initialized as a set of random vectors at the beginning and refined iteratively, as in [103]. Note that due to the attention mechanism, the queries are still refined based on the input image and are not static throughout. This is a promising direction for future work.

4-6 Issues with the P2P Localization Algorithm

In this section we explore the issues with the P2P localization algorithm, namely the inability to learn local feature matches and the tendency to learn the global vehicle location instead.

The reason for the poor localization performance of the P2P method is the general limitation of point matching, which stems from the fact that a single image is used to supervise all point matches within the image. Depending on the sampling type and density, the point clouds may contain up to $\sim$20 000 points each, each of which needs to be localized based on a single image.

Firstly, such a task is inherently prone to overfitting to noise, since due to occlusions and empty space there is not enough information at the scene to localize each 3D point. We would expect that points that are not possible to localize would simply be ignored by the model, since in expectation learning any bias for a point that is impossible to predict should be as good as random. However, due to the sheer number of points, different biases are learned during training, which can not generalize to the test set.

Secondly, the variable, which determines the ground truth location for each query ground point is the ground truth vehicle location. So if the model were to learn where the vehicle is with respect to the aerial image, it would help reduce loss for all $\sim$20 000 points. This is a quicker way to improve on the training objecting than to learn the matches for each individual local feature.

The issue is that the current P2P formulation allows the model to extract global information from the aerial image regarding the vehicle location and propagate it to all individual points. This behaviour is shown on Figure 4-8. Instead of retrieving the semantic descriptors for all points and matching based on image content, the aerial feature extractor learns the road prior, which provides the camera $\rightarrow$ world transformation, while the ground feature extractor makes use of the positional embedding to learn where each ground point is with respect to the vehicle. This way, the algorithm is cognisant of the location of each ground point with respect to the geo-referenced aerial image. This knowledge is then used to select the corresponding aerial match.

Even when the pixel embedding is excluded and only the depth coordinate is embedded, the model is able to learn the positional information from the border effects. Even when reflect padding is used instead of zero padding at all convolutional layers of the backbone, the model is still able to learn the positional information.

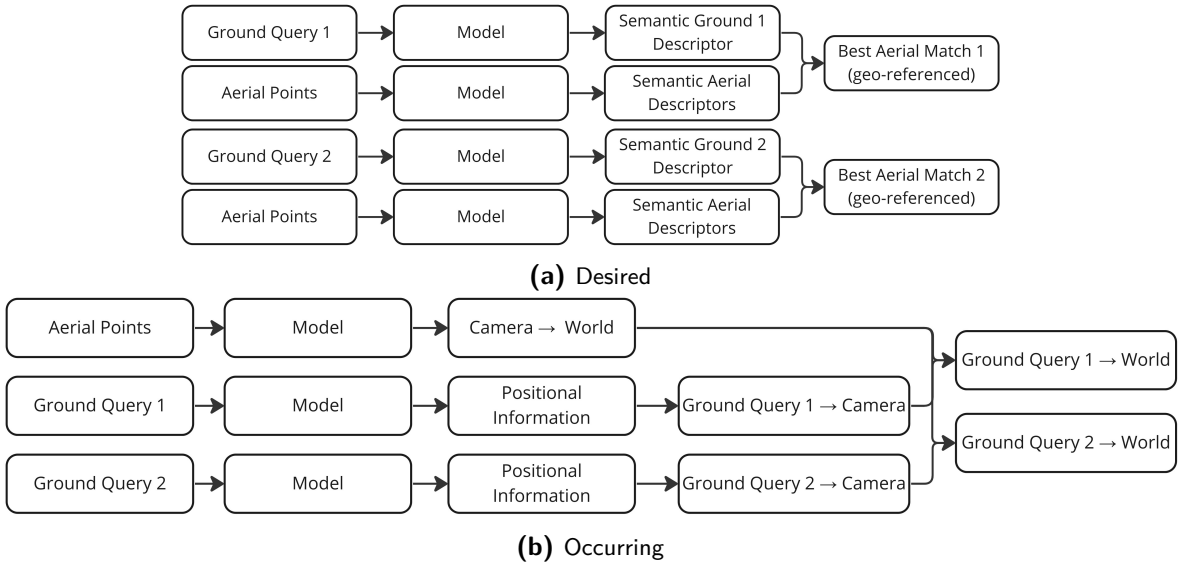


Figure 4-8: Desired and occurring behaviour of the P2P localization algorithm.

There are three cues, which justify that what is shown on Figure 4-8b is indeed occurring.

1. The error distribution follows the distribution on the vehicle location, rather thn semantic uncertainty on the local feature, as shown on Figure 4-9a.

2. Locations of impossible matches are predicted surprisingly well, as shown on Figure 4-9b.

3. Setting all color channels of the input ground image (i.e. making it black) at test time has a negligible influence on the local matching results, as shown on Figure 4-9c.

Unfortunately, there is no way to prevent the model from exploiting the road prior completely. Additionally, it is likely that the size of the dataset and the quality of the ground truth matches are aggravating factors, further emphasizing this limitation. If the dataset was larger, the model would still learn the road prior at first, but once it is exploited, the only way to make progress on the training objective would be to learn the local matches, given that overfitting is prevented via a larger dataset. Additionally, oftentimes the poor quality of the ground truth camera pose data results in incorrect supervision during training, as shown in Appendix C.

4-7 Ablation Study

In this section we demonstrate the results of the ablation study in order to justify the design choices made in chapter 3. The ablation study is guided by the ablation hypotheses, as defined in section 3-10. The overview of hypotheses and their corresponding sections is shown on Table 4-12.

(a) Error distribution follows vehicle location distribution. Even though the semantic content may be very discriminative longitudinally, the longitudinal error is still larger than lateral, since the vehicle location is more uncertain longitudinally.

(b) Impossible queries have good predictions. Here the query ground point is clearly behind a building, so it is impossible to localize it from the aerial image. Nevertheless, the model predicts the correct location.

(c) Setting color channels of the ground image to zero only has a slight influence on matches

Figure 4-9: Cues that the P2P algorithm is learning the vehicle location from the aerial image. Yellow arrow represents the vehicle location. Red points represent attention scores with weights proportional to opacity. Yellow star represents the ground truth location of the query point, which is depicted in cyan in the ground view.

Table 4-12: Overview of ablation hypotheses and corresponding subsections.

Method	Hypothesis Label	Hypothesis Description	Section
Regression	AH1	Extracting features before projecting them is more effective than projecting the original images.	subsection 4-7-1
Regression	AH2	Projecting the aerial image onto the ground domain is more effective than projecting the ground image onto the aerial domain, which is still more effective than not projecting at all.	subsection 4-7-1
Local Matching	AH3	The best positional embedding method is the one that embeds the most information, i.e. one that embeds u, v and d, where u and v are pixel coordinates and d is the depth of a 3D point.	subsection 4-7-2
Local Matching	AH4	Negative log likelihood loss is better suited for local feature matching, since local features are expected to be unimodal. Thus, reducing the variance of attention scores around the ground truth is also a good proxy for reducing the localization error.	subsection 4-7-2

Table 4-13: Mean error of the regression approach on "fine" setting of the KITTI dataset. The best result for each metric is highlighted in bold. A2G - Aerial to Ground, G2A - Ground to Aerial, Direct - No projection.

		A2G		G2A		Direct	
		RGB	Features	RGB	Features	RGB	Features
Train	Long. Error (m)	0.84	**0.46**	2.00	0.54	2.16	0.53
	Lateral Error (m)	0.37	**0.18**	0.26	0.20	0.26	0.21
	Orien. Error (deg)	0.73	0.37	0.32	0.43	**0.31**	0.45
Test 1	Long. Error (m)	1.07	**0.68**	2.05	0.75	2.22	0.73
	Lateral Error (m)	0.46	**0.25**	0.31	0.26	0.31	0.27
	Orien. Error (deg)	0.72	0.40	**0.32**	0.48	**0.32**	0.50
Test 2	Long. Error (m)	**2.16**	2.29	2.50	2.40	2.59	2.41
	Lateral Error (m)	1.30	1.23	1.35	1.31	1.51	**1.20**
	Orien. Error (deg)	0.71	0.83	**0.48**	1.73	0.52	1.75

4-7-1 Importance of the Projection Module

The quantitative performance of the regression approach is depicted on Table 4-13. This helps us to analyse the first two ablation hypotheses: **AH1** and **AH2**, namely that extractive features prior to projection is more effective than projecting the original images and that projecting the aerial image onto the ground domain is more effective than projecting the ground image onto the aerial domain, which is still more effective than using no projection at all.

The first hypothesis AH1 is largely true for positional errors - all but one of the errors are lower for the features than for the RGB images, and the outlier is largely insignificant, since the value is close to random anyway. Interestingly, however, orientation performance seems to be better for the RGB images than for the features. Noting that the orientation performance is best when the aerial image is not distorted through a projection (G2A and Direct), we can deduce that the model largely uses the aerial image to predict the orientation, without significant influence of the ground image. The road prior makes this possible.

The second hypothesis AH2 is also largely true, since the majority of the errors are lower for

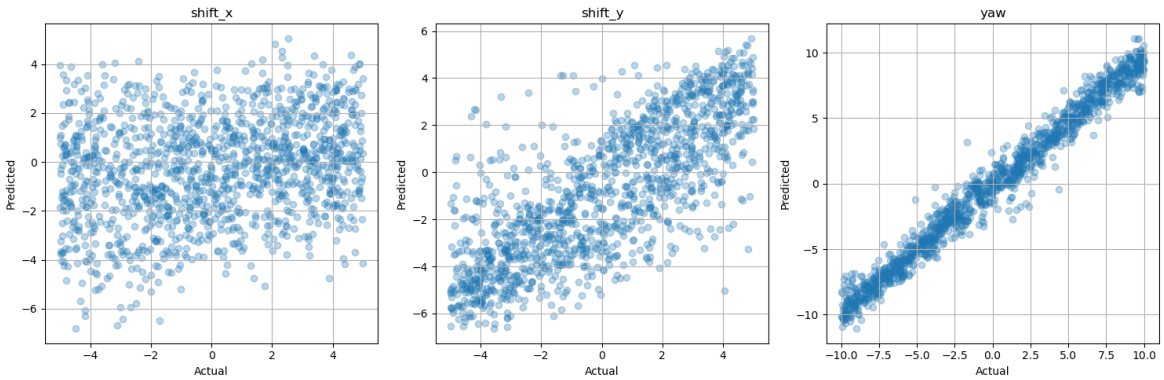


Figure 4-10: Predicted offset vs actual (ground truth) offset for the baseline approach. Left: longitudinal, middle: lateral, right: orientation. The reported values are in meters and degrees. A2G projection applied.

the A2G projection than for the G2A projection. Where this does not hold, the difference is typically small. This is in line with the intuition that transforming the aerial image still retains sufficient information for informative feature matching, while transforming the ground image does not.

Another interesting observation is the comparison to HighlyAccurate [33]. Evidently simply concatenating image descriptors and feeding them through an MLP is more effective in the fine grained setting than iterative refinement fed with the difference between the two descriptors, as was done by [33]. Especially the difference in orientation performance is significant. This is in line with the conclusion of HighlyAccurate that LM optimization is not effective in orientation estimation due to the lack of rotational equivariance in CNNs. Additionally, this is in line with the conclusion of Appendix B regarding the undesired behaviour of iterative refinement.

The surprising capacity of the proposed method to predict orientation is also depicted on Figure 4-10. Clearly, the model is able to predict the orientation better than position. Interestingly, unlike in the case of HighlyAccurate [33] (Figure A-4), the errors almost completely do not correlate with the ground truth offsets - the model is equally as capable to predict high and low deviations from the ground truth.

4-7-2 Optimal Positional Embedding Dimensionality and Loss Function

In this section, we present the results of tests to determine the optimal positional embedding dimensionality and the optimal loss function. The results are presented on Table 4-14.

Note that the values on Table 4-14 represent the point errors, rather than camera pose error. These errors are the point differences described in chapter 3. Therefore, it is expected that the lateral error is so large - after all both camera position and camera orientation errors contribute to the lateral error of each point. Nevertheless, this is a good indicator of the performance, since the same metric was used to train the network.

Also note that concatenating sinusoidal embedding was determined to performed best out of the options listed in the “Positional Encoding” section of section 3-6. Thus all results in

Table 4-14: Effect of positional embedding dimensionality and loss function on the performance of the P2P localization algorithm. The best result for each metric is highlighted in bold. Values represent mean absolute error.

		NLL, depth embedding only	NLL, uvd embedding	MSE, uvd embedding
Train	Longitudinal (m)	1.53	1.62	**1.35**
	Lateral (m)	3.21	**3.01**	3.15
	Altitude (m)	0.45	**0.42**	0.45
Test 1	Longitudinal (m)	**1.41**	1.44	**1.41**
	Lateral (m)	3.25	**3.10**	3.22
	Altitude (m)	0.57	**0.50**	0.56
Test 2	Longitudinal (m)	1.97	1.92	**1.87**
	Lateral (m)	3.58	3.61	**3.49**
	Altitude (m)	**0.43**	0.54	0.51

Table 4-14 are reported with concatenating the sinusoidal embedding.

Overall, all three method demonstrate similar performance. Regarding **AH3**, we can see that additionally using pixel coordinates in the depth embedding slightly improves performance. Regarding **AH4**, Negative Log-Likelihood (NLL) loss seems to perform slightly worse than its Mean Squared Error (MSE) counterpart. This is against our intuition that forcing matches to be close by should be a good proxy to finding a good match overall.

However, note that the difference is small. Furthermore, when inspecting qualitative results of each match, the fact that the MSE loss does not enforce unimodality is apparent, for instance as shown on Figure 4-11. On both Figure 4-11a and Figure 4-11b, the final output is similar, however, the attention scores themselves are different. The dispersed matches on Figure 4-11a are not a problem in MSE loss, since the loss is only applied on the weighted average of the scores. However, the NLL loss additionally ensures that the variance is small, which is desirable for interpretability and would be a good proxy if the model was to achieve better performance. Therefore, the optimal configuration is chosen with NLL loss and positional embedding with pixel coordinates.

(a) Example with MSE loss

(b) Example with NLL loss

Figure 4-11: Example of point matching with MSE and NLL loss. The yellow star is the ground truth location of the query point. The green arrow is the ground truth vehicle location. Red points represent attention scores with weights proportional to opacity. The purple triangle is the final prediction, after cross-attention.

Chapter 5

Conclusion

The introduction identified the vehicle localization performance as one of the leading causes for the slow adoption of truly autonomous vehicles. One study found that for autonomous vehicles to be as safe as commercial aviation today, the vehicle needs to localize itself with an accuracy of 10 cm to 20 cm with 95% confidence, depending on the vehicle and the road type.

Currently the vast majority of vehicles use GPS as their primary localization sensor, which is not only inaccurate, but also unreliable in urban environments. Alternatively, HD maps are also not a viable solution, since they are expensive to create and maintain, and are not available in all areas. Neither of these solutions can provide the required accuracy and reliability.

Therefore, following the corpus of literature in the field of visual localization, we identified Cross-View Geo-Localization (CVGL) as a viable alternative, which is concerned with determining the vehicle pose using the on-board camera feed and a geo-referenced aerial coverage of the area. CVGL can be further divided into Cross-View Image Retrieval (CVIR), which is concerned with finding the most similar aerial image from a database given a query ground image, and Cross-View Pose Estimation (CVPE), which is concerned with estimating the vehicle pose within a given aerial image. The latter is the focus of this thesis.

There is only a handful of works in the field of CVPE. Evaluating the literature, it has been discerned that not only do the state-of-the-art methods fall significantly short of attaining the required levels of accuracy, but the performance, particularly in the longitudinal direction is not far from random in some methods.

Other limitations include the poor motivation for the choice of the location prior and issues with interpretability, particularly when global features are used. Furthermore, due to the relatively small sizes of open-source datasets, the performance

Therefore, despite being challenging, creating a visual localization method, which is able to provide the required accuracy and reliability, while meeting the requirements of inference speed, interpretability and data-efficiency remains an open problem. In order to formality state the problem, we have formulated the following **Main Research Question (MRQ)**:

> **"How can Cross-View Geo-Localization methods using camera input alone provide localization performance sufficient for autonomous driving?"**

In order to answer the main research question, four sub-questions were posed:

SRQ1: "How can the localization setting be refined to tailor to the requirements of autonomous driving?"

SRQ2: "How can local feature matching be optimized for the fine-grained localization setting?"

SRQ3: "How can interpretability of CVGL be improved?"

SRQ4: "How can data efficiency of CVGL be improved?"

In chapter 2 we noted that existing methods either use global features entirely, or local features but with supervision at the camera pose level. This makes it difficult to interpret the way the local features influence the camera pose, nor is it even possible to confidently state that the local features are indeed local. Therefore, in chapter 3 we propose two methods that use local features.

In the first method, Point-to-Point (P2P) matching, the local features are projected onto two point clouds, one for the aerial and one for the ground image. Each point in the ground point cloud is then used as a query in the self-attention mechanism to determine its position based on the query aerial points. The model is supervised on the point-level. The pose is estimated such that the distances between the query points and corresponding predictions is minimized.

In the second method, Ray-to-Point (R2P) matching, the same methodology is used, but with ground ray set instead of the ground point clouds. The point distances as a optimization objective is replace with angular differences between the query ray and the angle that the predicted point makes with the camera location.

Furthermore, we hypothesized that despite the relatively poor localization performance of HighlyAccurate [33], the failure cases are specific to the coarse localization setting. HighlyAccurate [33] uses a projective transform, which merges the domain gap between the aerial and ground feature maps. The two feature maps are then subtracted to provide the optimization target for Therefore, we made some key adjustments, such as improving the backbone network and replacing the iterative pose estimation step with a regression step that allows for a 6-Degree of Freedom (DoF) camera pose output.

The answers to the four sub-research questions, followed by the answer to the main research question are provided below.

SRQ1: "How can the localization setting be refined to tailor to the requirements of autonomous driving?"

In chapter 2 we determined that the currently used localization setting with a position uncertainty of $20\,\mathrm{m} \times 20\,\mathrm{m}$ and an orientation uncertainty of 10° is not well justified. Originally,the

choice of the orientation prior was based on the CVIR literature, where central region in the retrieved aerial patch spans approximately $20\,\text{m} \times 20\,\text{m}$. However, such a justification is not reasonable in the context of autonomous vehicles in the real world, since it should be linked to some inherent uncertainty metric, rather than the choice of the aerial image patch.

In section 3-1 we propose the new localization setting, which is more fine-grained than the currently used one. We name the setting "fine". In the "fine" setting, the position uncertainty is $5\,\text{m} \times 5\,\text{m}$ and the orientation uncertainty is 10 degree, which was based on the assessment of GNSS errors as well as performance of existing state-of-the-art methods. We use this setting to design th localization methods.

SRQ2: "How can local feature matching be optimized for the fine-grained localization setting?"

One distinction of the "fine" localization setting with the existing settings is the fact that the initial positions of the local features are close to their ground truth positions. Thus, we can look for matches in a truly local neighborhood in 3D space, which was the motivation for modelling the problem as P2P and R2P matching.

Another distinction is the fact that neglecting the effects of pitch, roll and altitude changes is likely more detrimental in the "fine" setting. Modelling the entire 3D space by the P2P and R2P matching algorithms allows us to account for these changes.

In terms of quantitative performance, however, the local feature matching approaches were worse than the proposed Regression approach. There are reasons to believe that the proposed local feature matching methods would underperform, such as the lack of end-to-end camera pose estimation and the absence of a hyperparameter search. However, the qualitative results demonstrated that the largest issue was that the number of local feature queries was too large. This resulted in two issues: (1) the road prior influence and (2) the influence of "unmatchable" points on the camera pose estimation.

The road prior influence stems from the fact that given a dataset where the image was always taken from the road, it is simpler to predict where the image was taken from and propagate this information to all local features, rather than match all local features in a single forward pass. Any incremental improvements in predicting the vehicle location on the aerial image will result in a larger loss minimization than improvements in local feature matching, per feature. We have shown (section 4-3) that the P2P architecture is able to propagate the predicted vehicle location to all local features. However, this was resolved by R2P, through the use of a smaller number of queries and more complex, angular differences as the optimization objective.

While reducing the number of queries by R2P allowed to mitigate the road prior effect, the influence of "unmatchable" points/rays on the camera pose estimation was still present, as proven by the poor quantitative performance, despite the qualitative results being promising. The "unmatchable" rays are those where the ground truth is either occluded or is beyond the scope of aerial image. The hypothesis was that the attention mechanism would rid of the influence of such rays by assigning the same attention weight with all keys, thus effectively increasing uncertainty. However, due to the sheer number of "unmatchable" rays, the model overfit to those, thus making the camera pose estimation based on ray predictions and uncertainty scores unreliable.

To conclude, we found that if truly local features are to be used, it is difficult to optimize both the predictions and the confidences jointly. For any given query ground image, there are only a few points in 3D space that can be matched in a way that is discriminative to the camera pose location, e.g. road markings, building edges. We believe that pre-determining these queries in advance is a promising direction for future research.

SRQ3: "How can interpretability of CVGL be improved?"

Improving interpretability was one of the key reasons to use local features, supervised at the point level. In section 4-2 we showed how the R2P method can be interpreted on both the camera level (the hypothesis probability map) and the point level (the attention weights of aerial points corresponding to each query ray).

Furthermore, we showed that the R2P method performs exactly as we expected in the way that confident predictions drive the hypothesis probability map. This is a important property, since it uniquely allows us to interpret, rather than just present the model's confidence in the camera pose estimation. Furthermore, we showed that despite good local feature matching performance, the camera pose estimation can still be poor with respect to the localization setting. A large reason for this is that even well matched local features on the scale of the $100\,\mathrm{m} \times 100\,\mathrm{m}$ aerial image are not sufficiently accurate to discriminate the pose in the small area of the camera pose uncertainty.

However, interpretability comes at the cost of performance, since the method can no longer be end-to-end. Nevertheless, we believe that finding local correspondences is the right approach to the CVPE problem, not only due to improved interpretability, but also simply because local correspondences is the underlying task that the CVPE problem is trying to solve, so coating it with global descriptors or training the model end-to-end may not improve performance and only worsen overfitting.

SRQ4: "How can data efficiency of CVGL be improved?"

Supervising local features at the point level rather than the camera level is a way to increase data efficiency because a single image pair contains hundreds of local points to be matched. This is somewhat reflected in the quantitative performance of the models - the Regression approach overfits the most, while R2P overfits the least.

However, the performance of the Regression approach is still better than the R2P approach, even on the cross-area setting, which makes it difficult to conclude that the Regression approach is truly more data efficient. If, as we discussed earlier, the R2P approach intrinsically predicts the camera pose first and then propagates it to the local matches, then there are clearly no improvement to data efficiency. Nevertheless, we have shown in section 4-2 that R2P is likely trying to match local features directly rather through than predicting the camera pose first. Overall, the data efficiency of the local feature matching approaches is inconclusive.

MRQ: "How can Cross-View Geo-Localization methods using camera input alone provide localization performance sufficient for autonomous driving?"

Following the study of the related works in chapter 2, we have identified that the existing state-of-the-art methods are far from being able to provide the required accuracy and reliability for autonomous vehicles. For instance, the 95% of the location predictions for safe autonomous operations should fall below a 10 cm to 20 cm error. However, the current state-of-the art can contain mean errors of up to 2 orders of magnitude larger. Due to the size of this gap, we propose a new localization setting, significantly more fine-grained, where initial uncertainty in the camera pose is driven by GPS errors.

In this thesis work, we attempt to improve localization performance in the fine-grained setting using local feature matching. We proposed two methods, which are supervised at the local feature level, rather than the camera pose. Apart from performance improvement, the design of the architecture was driven by the need for interpretability (to improve the understanding of the model's behavior) and data efficiency (given the small sizes of available datasets). We additionally adapted an existing approach, HighlyAccurate [33] to the fine-grained localization setting by addressing its shortcomings. We call it the Regression approach.

We found that quantitatively the Regression approach outperforms the local feature matching approaches. However, the Regression approach has more overfit due to the use of global features (thus worse data efficiency) and less interpretability. We found that uniquely not predicting the depth of the local features, but rather matching them based on angular errors is a promising direction for future research.

To address the **MRQ**, through the novel localization setting we have bridged the gap between existing and required localization performance. Despite quite promising qualitative results, the quantitative performance of the proposed local feature matching approaches is still far from the required accuracy and reliability. The main reasons identified for this is the influence of the local features that cannot be accurately matched on the training procedure. Other reasons include the relatively small sizes of available datasets and the poor quality of the ground truths (Appendix C), which is especially detrimental in the fine-grained setting.

Future Work

Given the promising qualitative results of the local feature matching approaches, we believe that the shortcomings of the current formulation are worth addressing.

First of all, there is a need to pre-select the local features that are worth matching. We have shown that predicting both the matches to the queries and controlling the confidences of these matches, defined by the weighted variance of matches is difficult to do jointly, more so if not performed end-to-end. Therefore, a promising direction is to perform keypoint detection prior to initializing ground rays/points. A similar conclusion was reached by [104], who published an update to HighlyAccurate, which involves sparse descriptors associated with ground-view keypoints. This would improve both performance and interpretability.

Secondly, it has been established (section 1-3) that due to the forward-facing cameras on the KITTI dataset, the longitudinal performance is significantly worse than lateral. This is because in the absence of a highly accurate depth prediction module, features directly in front

of the vehicle are only discriminative in the lateral direction. Similarly to the best performing works in the field [39, 104], we believe that using multiple cameras, including side and rear cameras, would improve performance significantly.

Finally, Appendix C identified multiple errors with the ground truths both on the large and small scales. These are extremely detrimental to the fine-grained setting and are likely the reason that some of the most recent works in the field [94, 104] do not report results on the cross-area setting of the KITTI dataset entirely. Therefore, we believe that the ground truths must be refined using at least the methodology presented in [39] and the proposed methods should be re-evaluated on the refined dataset.

Appendix A

In-Depth Review of Key Cross-View Pose Estimation (CVPE) Methods

In this appendix, the methods mentioned in section 2-3 are presented in detail. For each method the methodology and analysis is performed.

A-1 HighlyAccurate [33]

A-1-1 Methodology

In [33], the authors utilize a differentiable non-linear least squares solver to iteratively refine the pose in three steps.

Unlike [82], in [33] the authors extract the feature maps from both images using a U-Net architecture. They retain the resulting feature maps from the last three layers, similar to [39]. Instead of fusing the feature maps, feature maps at different resolutions are used to predict the camera pose at different precisions, as shown on Figure A-1.

At each feature depth level, a three-step optimization is performed. At each step the aerial features transformed into the ground domain are subtracted from the ground feature. Minimizing the resulting vector is the goal of the Levenberg-Marquardt (LM) algorithm, which was chosen as the non-linear least-squares solver. The output of each step of the LM optimization is a pose offset. This offset is applied on the previous pose and the resulting pose is used as the new point at which the perspective transform is applied to generate a new feature map. Once the three steps are exhausted, a higher-resolution feature map is used instead and the steps are repeated. Once all three feature maps have been exploited for optimization, the resulting pose is the model's output.

There are a few important takeaways from both the approach and the conclusions of the paper. Due to the relevance of this method to the potential fine-grained research, we discuss these in detail below.

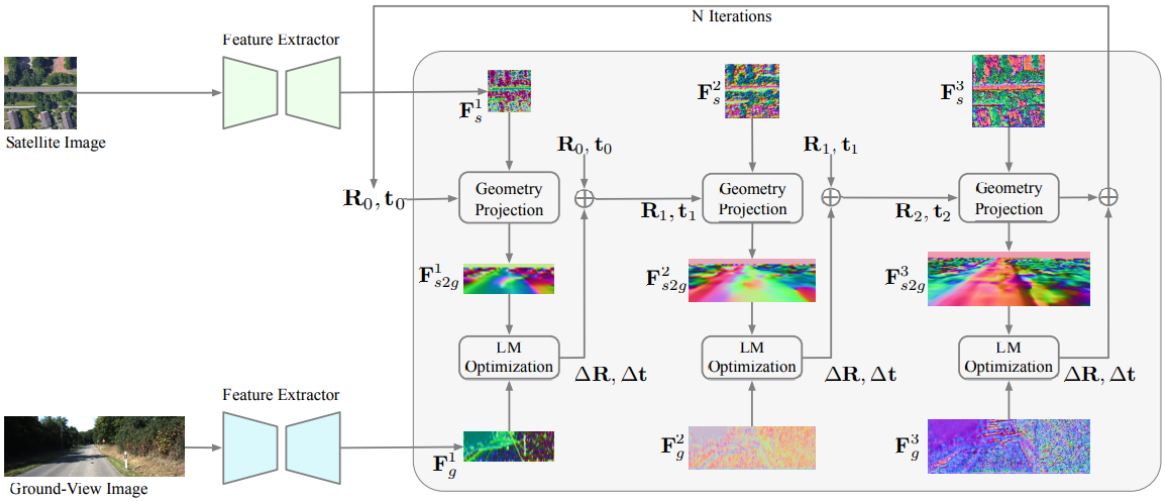


Figure A-1: Iterative pose estimation from [33]

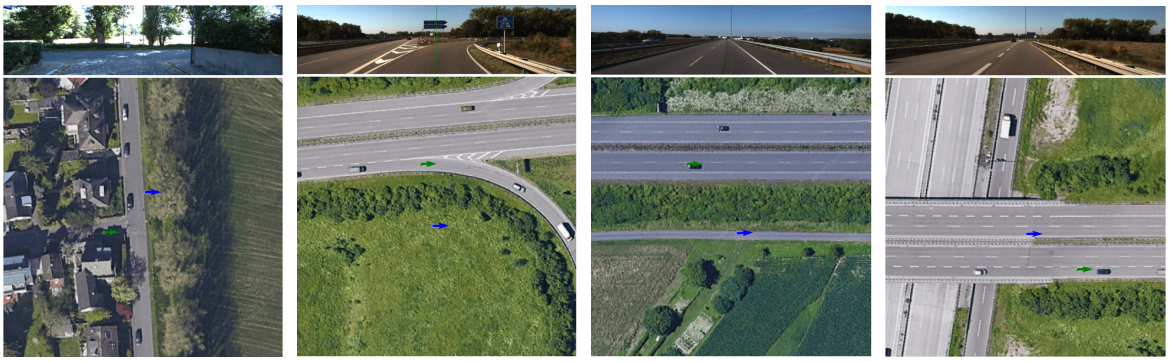
Figure A-2: Failure cases from [33]. Red arrows represent initial poses, green arrows denote final predicted poses, and blue arrows indicate GT poses. The magenta points pinpoint intermediate locations during optimization.

A-1-2 Analysis

Globally, the proposed methodology seems as though it should be more suited to a fine-grained task, similar to what this thesis work is concerned with. Using an iterative optimization algorithm, such as LM necessitates the assumption that the feature distribution is both unimodal and differentiable.

The failure cases of the model reported by the authors are shown in Figure A-2. There, the authors simply demonstrate that the model is unable to localize longitudinally. They state: "The scene facades shown in the ground-view are not visible in the overhead view. We can only determine the lateral translation of a ground vehicle."

While it is true that localization in the longitudinal direction is difficult, in my opinion the real weakness of the model is when the features are either not unimodal or not differentiable, such as those shown on Figure A-3. On Figure A-3a and Figure A-3b the initial position is too far from the ground truth such that moving a little closer to the ground truth does not make the features match better. Because of this the gradient in the direction of the ground truth is not positive, which means that the model is unable to converge to the ground truth. On Figure A-3c and Figure A-3d the initial position is in between two modes of the feature distribution. In those particular cases, the gradient was stronger in the direction of the wrong mode, which is why the model converged to the wrong solution.



(a) UD, ex.1 **(b)** UD, ex.2 **(c)** MM, ex.1 **(d)** MM, ex.2

Figure A-3: Failure cases from [33], not reported (own work). UD: uninformative derivatives, MM: multimodal. The initial guess is the image centre. Blue - final model output; green - GT.

Multimodality and uninformative derivatives, however, are largely influenced by the quality of the prior available. If the prior is weak, samples such as those depicted on Figure A-3 will occur more frequently. On the other hand, Figure A-3b for instance could never occur with a ± 5 meter prior.

This feature of iterative methods can also be visible on Figure A-4. It is natural that as sample difficulty increases, the error increases as well. However, note that when the shift from the centre is > 20 meters, the error is never 0. The model is never able to localize correctly when it needs to iteratively predict the location more than 20 meters from the initial guess. This is because the derivative in the correct direction is not high enough, since the rate of change of resemblance of the two feature maps is too low when shifted in the right direction.

Due to the fact that even despite these significant failure cases, the model performs quite well on the large-scale dataset. There are reasons to believe that iterative pose estimation is more suitable for fine-grained localization than other methods. Specifically because we can expect these failure cases to be less pronounced in the fine-grained localization domain, i.e. in a fine grained setting only the left hand side of Figure A-4 becomes relevant.

Regarding the raw performance of the model in the $40\,\mathrm{m} \times 40\,\mathrm{m}$ uncertainty window, however, the comparisons with UncertaintyAware [39] are not so clear-cut. The only common dataset that the models were compared on are the first two scenes of the FordAV dataset [69]. Here, HighlyAccurate [33] is inferior in terms of recall at all reported thresholds by a large margin, see Table A-1.

Interestingly, HighlyAccurate demonstrates similar longitudinal localization performance as UncertaintyAware when the latter does not use vehicle frames (so when the ground image is blacked out during training). At first glance, this may seem as though HighlyAccurate was not able to learn anything more than the road prior for longitudinal localization.

However, the road prior can only be exploited when the aerial image is directly utilized. Because HighlyAccurate transforms the aerial image using a projective transform, the global scene information is lost. After the projection, during feature subtraction the model has no way of knowing where the projection was taken with respect to the aerial image. Therefore, there are reasons to question the conclusion presented in [39]:

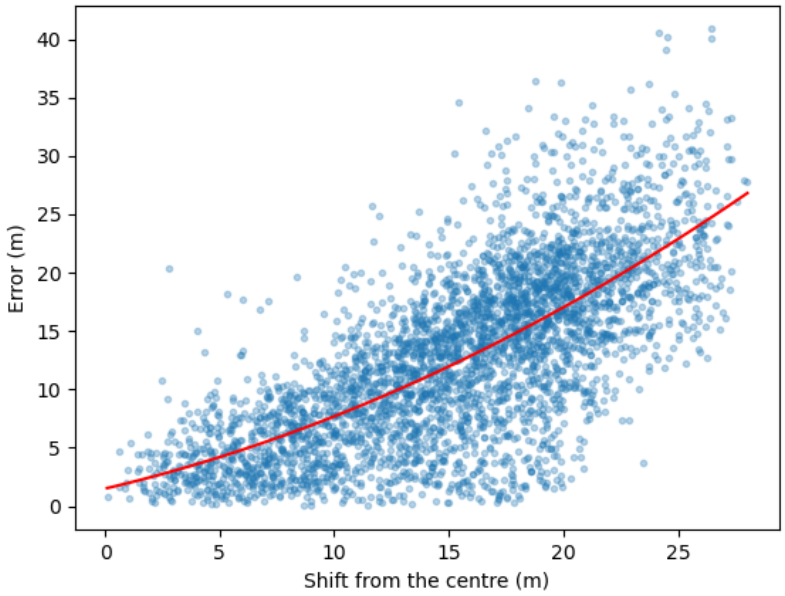


Figure A-4: Effect of sample difficulty on error for HighlyAccurate [33]. Since the centre represents the initial guess, samples that are far from the centre need to be iteratively refined a lot to reach the ground truth. Red line is a polynomial fit. Own work.

	Cross-area	Cross-vehicle	Log 1						Log 2					
			Lateral			Longitudinal			Lateral			Longitudinal		
			1.0m	3.0m	5.0m	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m
HighlyAccurate	✗	✗	46.1	70.4	72.9	5.3	16.4	26.9	31.2	66.5	78.8	4.8	15.3	25.8
UncertaintyAware	✗	✗	**96.3**	**99.6**	**99.6**	**76.0**	**95.3**	**96.0**	**88.0**	**99.9**	**100.0**	**58.9**	**93.3**	**93.6**
UncertaintyAware	✓	✓	77.0	96.2	97.6	24.0	67.6	76.1	73.0	96.5	97.8	25.6	61.7	69.4
UncertaintyAware w/o vehicle frames	✗	✗	15.1	1.3	72.0	5.0	15.2	24.4	11.3	37.8	62.2	4.7	15.3	26.0

Table A-1: Quantitative comparison between HighlyAccurate [33] and UncertaintyAware [39]. Recall in percent on the first two scenes of the FordAV dataset is shown. Results provided by each paper. Initial pose is chosen in $40\,\mathrm{m} \times 40\,\mathrm{m}$ around the vehicle with up to $20°$ of rotation noise. Own work.

> "The model [w/o vehicle frames] shows a performance similar to the previous state-of-the art HighlyAccurate [33] for longitudinal recall indicating that their model might rely mainly on prior poses w.r.t. the aerial image rather than on cross-view matching"

Nevertheless, the fact that it is difficult to achieve good longitudinal performance with a single, limited-FoV needs to be noted, as shown on Table A-1. Longitudinal localization is mostly performed using features to the sides of the vehicle, so a lack of such features as a result of the limited FoV is detrimental to performance.

We can also separately analyze position and orientation performance. Typically we would expect the positional performance to be better when feature matching occurs in the ground domain, while orientation performance should be better in methods that match features in the aerial domain. This is because a small rotation will be virtually unnoticeable in the ground view. In the aerial view, however, such a rotation will deform the feature maps significantly at a large depth. Positional error, on the other hand, should be lower when matching occurs in the ground domain, because the pixels in the ground image that are close to the vehicle have a higher resolution. So even small offsets in the lane markings nearby are easily detected.

Interestingly, UncertaintyAware [39] were not able to confirm this. For some reason, UncertaintyAware, where we would expect good orientation performance, do not report any orientation results at all. HighlyAccurate [33], on the other hand, do report orientation performance, but it does not look promising. On the second scene of FordAV, for instance, the orientation performance is completely random. Where the recall at 1° for a random model would be 10%. the recall of HighlyAccurate model is 9.74%.On KITTI test1 and test2 datasets the orientation performance is twice better, but is still substandard for autonomous vehicle localization.

Finally, the choice of the optimizer used by HighlyAccurate is interesting. The authors perform an ablation study, where they show that LM optimizer performs best when compared with SGD, ADAM, and a network-based optimizer. They claim [33] that the adaptive second-order LM optimization is "essentially guaranteed to find at least one local minimum of a cost function", while SGD and ADAM can not. This was backed by the ablation study, as the lateral performance indeed improved significantly when LM was used.

The comparison with the network-based optimizer is less convincing. The network-based optimized consists of a set of convolutional and some fully connected layers. The network-based optimizer performed slightly worse in terms of positional performance but more than doubled orientation recall at 1°. The authors explain this with the fact that regular CNNs are not translation-invariant, so even a small change in orientation angle will lead to a big difference in the CNN feature maps.

If this explanation is correct (which is questionable, since a ground image orientation is similar to lateral translation, other than the small non-linearity associated with perspective effects), two conclusions can be drawn from this. Firstly, while it takes a lot of time to create a differentiable LM optimizer for a problem such as iterative camera pose estimation, the network-based optimizer is simple to test with and offers almost similar performance. This is useful with regards to the subsequent thesis work. Secondly, if the problem with a CNN-based optimizer is the translation invariance, perhaps a transformer-based optimizer is a better fit for this task.

A-2 UncertaintyAware [39]

A-2-1 Methodology

First, ground and aerial features are extracted using a pre-trained ConvNext backbone [105]. Since [106] have shown that most modern object detection frameworks benefit from multi-scale attention maps, the authors of [39] also extract multi-scale features with different resolutions and bi-linearly interpolate them to the same size. The feature maps are then summed and processed by a small Multilayer Perceptron (MLP) to obtain the final feature representation.

Once the final aerial and ground feature representations are obtained, the BEV representation is initialized with learnable parameters and iteratively refined by attending to the ground features. The iterative refinement is two-staged. In the first stage, cross-attention is performed where Bird's-Eye View (BEV) grid cells attend to ground image features and to gather ground information into the BEV representation. Only the ground features that could be projected nearby based on camera intrinsics are used as queries, while far-away ones are discarded.

To do so, the authors need to sample correct ground features at each BEV query. The BEV cells are first lifted to 3D using a pillar of z points uniformly sampled from h_{min} to h_{max}. The points are then projected onto the ground image using camera intrinsic and extrinsic parameters. However, this means that the BEV features contain only a sparse set of features from the ground image.

In order to incorporate fine-grained positional information despite having a rather coarse set of value tokens in the ground view, the authors make use of deformable offsets introduced by [107]. For each 2D BEV cell B_{xy}, offsets $\Delta p_j \in \mathbb{R}^2$ with $j \in \{1, \ldots, z\}$ are predicted using z linear transformations on B_{xy}. Thus a total of z offsets are predicted per BEV pillar, one per height bin. The features at each height index j and then projected according to Δp_j and sampled using bilinear interpolation as

$$f_{ij} = F_{Gi}\left(\frac{p_{ij} + \Delta p_j}{s_G}\right) \tag{A-1}$$

where F_{Gi} is the ground feature map, i is the index of the camera, $p_{i,j}$ is the ground pixel coordinate for camera i for the point at height j and s_G is the ground feature map stride. In other words, the feature for i-th camera view and j-th point in the pillar is obtained by sampling the ground feature map at the location $p_{ij} + \Delta p_j$. The resulting feature serves as value in the cross-attention module.

Note that offsets are predicted based on BEV cell location and per height bin, which gives the model the 3DoF freedom to project the 3D BEV voxel around the ground image. The offsets are also predicted in pixel coordinates, which allows to incorporate perspective effects directly.

After the cross-attention stage, the authors perform self-attention on the BEV features to refine the BEV representation. This is done using a single layer of SegFormer [108]. SegFormer uses a spatial-reduction attention, to reduce the computational complexity. When used as queries, BEV tokens are used in the traditional way. When used as values, however, the spatial resolution of tokens is first reduced by a convolution with some stride.

Overall, the iterative refinement stage, consists of iteratively applying cross- and self-attention as described above. Through the ablation study, the authors found that 3 iterations are sufficient to achieve the best performance, although the mean error is increased by only 5 cm (from 1.19 m to 1.24 m) when only one refinement block is used, which somewhat questions the need to refine the BEV representation iteratively.

After the BEV representation is refined, the authors center the BEV representation at each candidate location and compute correlation with the aerial feature map. These correlations are then used to compute the likelihood of each candidate location. The candidate with the highest likelihood is chosen as the final prediction.

A-2-2 Analysis

Despite the impressive results on the cross-area set, the paper has some limitations. First of all, the authors use a camera rig with 6 cameras, whereas some of the other methods use a single camera. This is especially beneficial for the longitudinal performance.

Secondly, the authors do not train the model on the KITTI dataset. The model is trained on a combination of five datasets: Argoverse V1, Argoverse V2, Lyft L5, Nuscenes and Pandaset. There are over a million samples combined across datasets, compared to just under 20 000 in KITTI. This makes direct comparisons with other methods difficult.

Thirdly, the authors generate pseudo-labels for all train datasets, as shown on Figure A-5. They do so by training a variant of their model, where instead of the ground image they use LiDAR data projected on the ground plane. They train the model on a subset of manually aligned samples. The predictions are used as ground truths to train the original model. The authors find that without these pseudo-labels the localization error doubles from 1.19 m to 2.37 m. Such ground truths were not available for other methods, making purely methodological comparisons difficult.

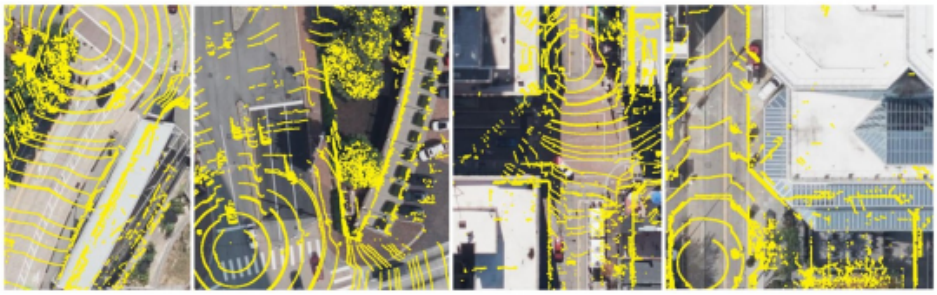
Lastly, as a critique applicable for any aerial view method, the solution captures a large prior that the vehicle is on the road. This is a simple way to improve performance on the relevant datasets, since learning the road structure from an aerial image is a trivial task for all widely-used backbones nowadays. However, such a prior can be detrimental in real-world applications, where detecting an off-road vehicle is precisely why localization is needed in the first place.

A-3 CCVPE [35]

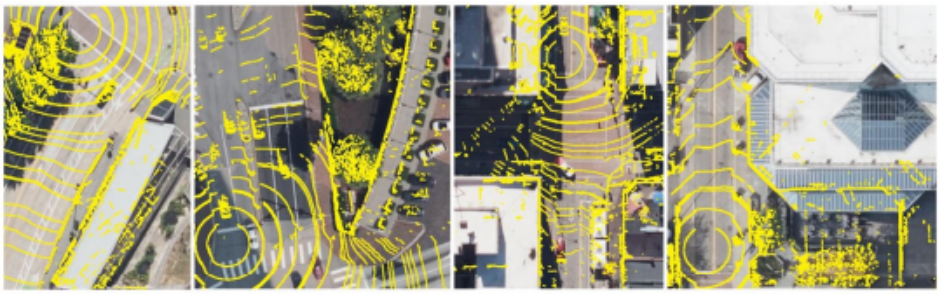
A-3-1 Methodology

Similar to [82], [35] agree that the image retrieval step may be omitted due to the strong GPS prior available. Thus, they focus on the pose estimation within the correctly matched satellite image.

In [35] the authors make use of the orientation information explicitly. The modification required to achieve this is a circular ground and aerial descriptors. First, the image descriptors are retrieved using a pre-trained VGG-16 backbone.



(a) Original ground truth poses



(b) Pseudo-labeled ground truth poses

Figure A-5: Comparison of the original ground truth poses and the pseudo-labeled ground truth poses. From [39]

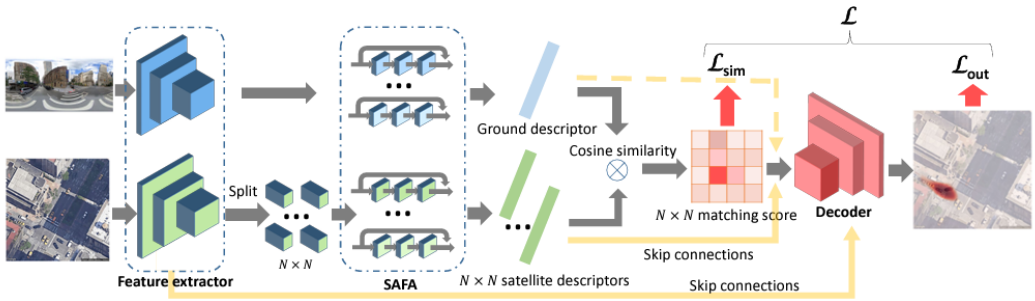


Figure A-6: Overview of the CVGL architecture proposed by [38]

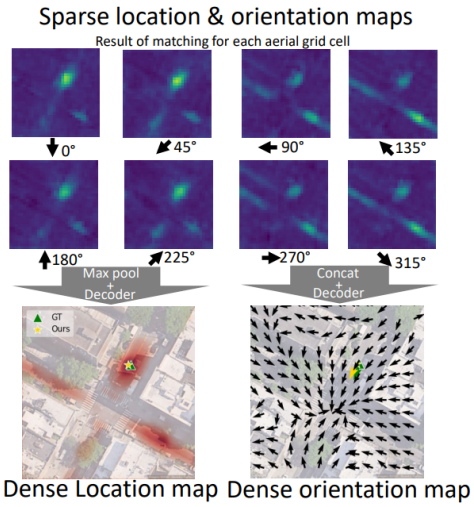


Figure A-7: Location heatmap and orientation map from [35]

The ground image is then divided vertically into cells. Each cell has an associated cell descriptor. Since the ground image is a panorama, the descriptor can be viewed as a circular vector, where the first and the last elements are adjacent.

The aerial image is geographically-aligned, such that each grid point in the aerial image is associated with a circular descriptor that preserves the orientation information (i.e. the descriptor has e.g. North-aligned and East-aligned components). Afterwards, the descriptors of each aerial grid point are matched with each sequential transposition of the ground descriptor. This way, the orientation information is explicit in the model architecture.

The maximum cosine similarity per transposition at each grid point is taken as the matching score for that grid point. This way, a probability heatmap is generated using a location decoder. At the same time, maximum pooling is used to predict an orientation at each aerial grid point. The orientation heatmap is generated using the orientation decoder. Both decoders also are fed with the aerial feature map, which helps capture the road prior.

A-3-2 Analysis

This modification further constraints the geometric correspondences between the feature vectors and achieves better performance, further reinforcing the need for explicit geometric constraints when performing feature matching in a shared representation. Another lesson is on how important it is to be able to deal with multi-modal distributions. As the authors note:

> "when the aerial view contains a symmetric scene layout, e.g. at crossroads, single-mode regression based methods [81], [85] might regress to a midpoint between visually similar locations, and optimization-based methods [33] might get stuck at a wrong local optimum."

It would be interesting to see if this conclusion will hold for a more fine-grained localization task, where the scene layout is less likely to be symmetric. More in-depth analysis of the method is present in the next section, where the method is compared with SliceMatch [34].

Table A-2: Quantitative comparison between SliceMatch [34], CCVPE [35] and HighlyAccurate [33] on the KITTI dataset. Results provided by each paper. Initial pose is chosen in $40\,\mathrm{m} \times 40\,\mathrm{m}$ area.

Same-Area		↓ Localization (m)		↓ Orientation (deg)	
		mean	median	mean	median
±10 deg	HighlyAccurate [33]	12.08	11.42	3.72	2.83
	SliceMatch [34]	7.96	4.39	4.12	3.65
	CCVPE [35]	**1.22**	**0.62**	**0.67**	**0.54**
No orientation	HighlyAccurate [33]	15.51	15.97	89.91	90.75
	SliceMatch [34]	9.39	5.41	**8.71**	**4.42**
	CCVPE [35]	**6.88**	**3.47**	15.01	6.12
Cross-Area		↓ Localization (m)		↓ Orientation (deg)	
		mean	median	mean	median
±10 deg	HighlyAccurate [33]	12.58	12.11	3.95	3.03
	SliceMatch [34]	13.50	9.77	4.20	6.61
	CCVPE [35]	**9.16**	**3.33**	**1.55**	**0.84**
No orientation	HighlyAccurate [33]	15.50	16.02	89.84	89.85
	SliceMatch [34]	14.85	11.85	**23.64**	**7.96**
	CCVPE [35]	**13.94**	**10.98**	77.84	63.84

A-4 SliceMatch [34]

A-4-1 Methodology

The third example of such feature manipulation is SliceMatch [34]. The authors use the azimuth prior - the idea that a vertical column on the ground image should correspond to a slice on the corresponding aerial view.

To do so, first the features are extracted from the aerial and ground images using a convolutional backbone (VGG [109] or ResNet [110]). Afterwards, the ground features per slice are summarized using a ground feature aggregator, which uses self attention to re-weight salient features. This is done to minimize the effect of the non-informative features, such as the sky, on the final ground descriptors. The ground slice descriptor is then found by averaging normalized features.

The aerial slice descriptor uses the ground features to guide feature aggregation via a cross-attention mechanism, where aerial slice descriptors attend to each of the ground slices to produce similarity score maps, one for each ground slice. These maps are concatenated along the channel dimension to the the aerial features. The resulting descriptor is re-weighed to produce the final aerial slice features maps of shape $N \times L \times L \times C$, where N is the number of slices, L is the spatial dimension of the feature map and C is the number of channels.

Finally, to predict the camera pose, the authors compare the aggregated aerial features at each 3DoF hypothesis with the ground features via cosine similarity. During training the task is framed as contrastive learning, where the correct hypothesis serves as the correct positive sample and the rest of the hypotheses are negative samples. For the loss function adapted infoNCE loss is used, where the effect of negatives is scaled via a hyperparameter.

A-4-2 Analysis

The results of the quantitative comparison between SliceMatch, CCVPE and HighlyAccurate are presented in Table A-2. Unfortunately, UncertaintyAware [39] did not report any results on the KITTI dataset, so it is not directly comparable.

Clearly, SliceMatch [34] only outperforms other methods on the orientation estimation without any orientation prior. This can be explained by the larger focus on orientation performance compared to localization in SliceMatch. At test time SliceMatch uses 15×15 location hypotheses and 64 for orientation. During training, the values are lower, but the ratio is almost identical. On the other hand, CCVPE [35] uses dense classification for position, i.e. for each of 512×512 pixels and 20 orientation bins. It is clear that SliceMatch targets orientation estimation and CCVPE prioritizes position.

When the 10° orientation prior is available, however, CCVPE outperforms the other two methods by a large margin on both localization and orientation estimation. In fact, when the search area is limited to $40\,\text{m} \times 40\,\text{m}$, a localization algorithm which always predicts the centre of the search area will have a mean error of 14.2 m. When the predictions are randomly sampled within the search area, the mean error is 18.9 m. Of course, any localization algorithm will start with the 18.9 m error, so the performance improvements are significant to the authors.

Still, as is evident that in the cross-area setting performance, CCVPE is the only method that is significantly better than 14.2 m error of the centre-predicting baseline. Even so, Table A-2 is a little misleading as it combines both longitudinal and lateral error. As Table A-1 has shown, low lateral error is easier to achieve than low longitudinal error due to the road prior. Thus, it would be interesting to see the longitudinal performance of methods shown on Table A-2, as it is a setting where road prior is not helpful.

The road prior is particularly influential in the case of orientation estimation of CCVPE. One example is shown on Figure A-8. The model has learned that cars drive on the right side of the road and since "the final orientation prediction is conditioned on localization, i.e. it is selected at the predicted location in the dense orientation map." [35], the model will predict the orientation that matches the orientation of the traffic. But suppose the vehicle overtakes another vehicle and drives on the left side of the road or due to a malfunction it significantly deviates from the road. The goal of the localization system on the vehicle is to precisely detect such situations and having the road prior is detrimental in such scenarios.

These results point at how difficult cross-view matching is across areas. The fact that the performance of the state-of-the-art methods is still far from perfect, even when the orientation prior is available, shows that the problem is far from being solved.

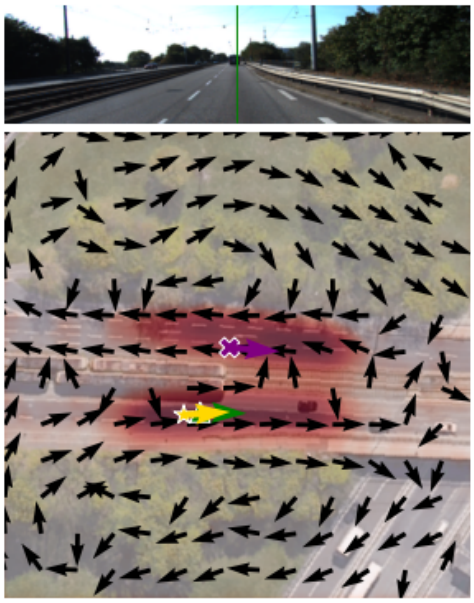


Figure A-8: Effect of the road prior on CCVPE performance from [35]. The green triangle is the ground truth, the yellow star is the CCVPE prediction and the purple cross is the HighlyAccurate prediction. The HighlyAccurate prediction uses a 10° orientation prior, while the CCVPE prediction does not.

Appendix B

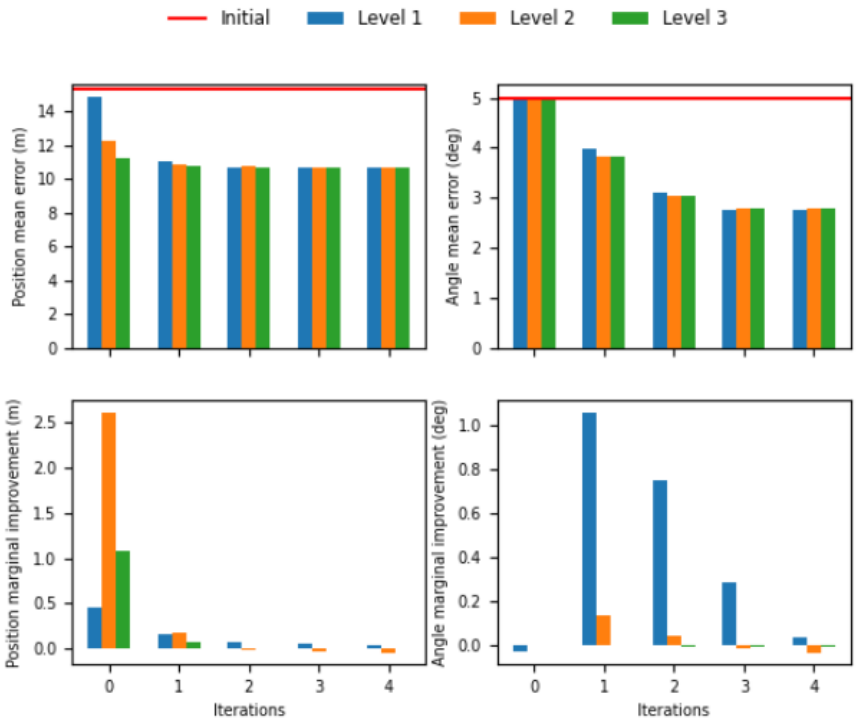
Methodological Failure of HighlyAccurate [33]

Upon studying the way the pose is iteratively refined by the HighlyAccurate [33] model, it was found that there is a strong mismatch with what the authors claim. HighlyAccurate refines the pose with five iterations, each of which has three levels. At each level feature maps with different resolution are used. The authors claim that first the coarse, low-resolution feature maps determine the approximate pose at Level 1, then the medium-resolution feature maps refine the pose at Level 2, and finally the fine, high-resolution feature maps at Level 3 make the pose more precise in the local vicinity, when the pose is already almost correct. The process is repeated for 5 iterations.

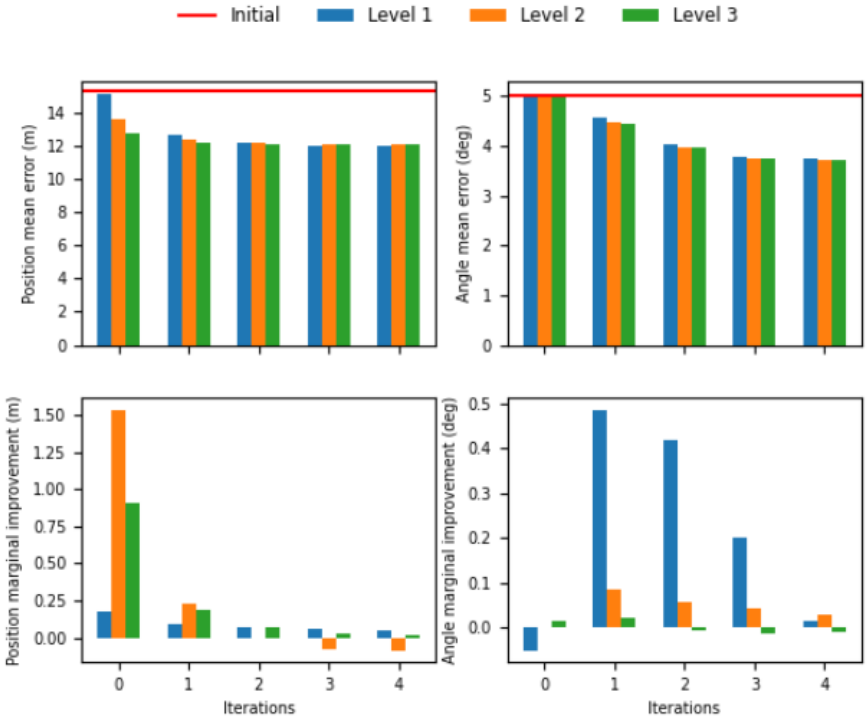
However, when studying how the error behaves during the iterations, it was found that the intuition is largely not correct, as shown on Figure B-1. For both the model we trained and the one provided by the authors, the positional error clearly only decreases significantly during the first iteration. The remaining iterations offer a negligible improvement, sometimes even making the error worse. In terms of the orientation error, only iterations 1, 2 and 3 seem to have a significant effect. Furthermore, in terms of the levels, the first level has a large impact on orientation and the second level on position. The third level has some effect on position.

Clearly the model learned to optimize the pose in a different way than the authors had intended. The marginal improvement does not uniformly decrease with iterations and levels. The fact that the model does not behave as intended questions the necessity of the iterative refinement. Optimizing the pose with a single pass, perhaps still making use of multi-scale features, would make for a less constrained optimization problem. In the current configuration, it is possible that each iteration/level refinement improves performance in spite of, rather than because of the previous one.

Additionally, in the original paper the authors perform an ablation study, where they attempt to reverse the order of iterations and levels and estimate the pose in a truly coarse-to-fine setting, such that first the five pose estimation iterations are performed at Level 1, then at



(a) Author-provided model



(b) Trained model

Figure B-1: Error dynamics of HighlyAccurate [33] model. Errors are average error values across the test1 set. Own work.

Level 2 and finally at Level 3. Such a configuration is a better representation of the intuition of the authors, since it is difficult to justify a coarse level refinement succeeding a fine-level one, as is done on Figure B-1. However, this modification resulted in a deterioration of performance, which was not explained by the authors. This is another indication that the model does not behave as intended.

Appendix C

Poor Ground Truth Quality in Processed KITTI Dataset

In this appendix we demonstrate that, despite attempts to manually filter the samples with inaccurate ground truth by HighlyAccurate [33], the resulting processed dataset is still quite inaccurate, which is especially detrimental in the fine-grained setting. We show that the poor ground truth quality is not only the case on the large scale (section C-1), but also on the small scale (section C-2).

C-1 Large-Scale Ground Truth Errors

Some errors are clearly visible without any further analysis. Examples on the test sets are given on Figure C-1 and on the training set on Figure C-2. These errors are extremely significant, especially in the fine-grained setting. In addition, train and test2 datasets are sequential, which allows to make a conclusion that not only are the individual samples inaccurate, but the errors are present in several neighboring samples. For instance, for train sample 10770, shown on Figure C-2c, the "ground truth" starts diverging from the actual trajectory at sample 10755 and continues up until sample 10878. This means that at least 130 samples are extremely incorrect just in this specific path segment.

The results presented on Figure C-1 and Figure C-2 are output by the dataloader proposed by HighlyAccurate [33] and used at least by [33, 34, 35]. Since KITTI dataset is used widely in the Visual Odometry (VO) literature with centimeter-level accuracy, we believed that the dataloader API is at fault, rather than the ground truth poses.

After further inspection of sample 10770, shown on Figure C-2c, it was unexpectedly found that the ground truth poses in the KITTI dataset are indeed incorrect. The associated

```
2011_09_30/2011_09_30_drive_0028_sync/oxts/data/0000000894.txt
```

(a) Test2, sample 1217. Clearly the camera is in the middle, if not to the right of the middle lane. The ground truth is almost at the left lane marking. If we assume the average lane width to be $3.5\,\mathrm{m}$, the lateral error on this image can be up to $1.75\,\mathrm{m}$, which is extremely large for the fine-grained setting.

(b) Test1, sample 307. The ground truth is on top of the building.

(c) Test1, sample 982. The ground truth is at the patch of grass to the left of where it should be.

Figure C-1: Examples of large-scale ground truth errors on the test sets. Green arrow represents the ground truth vehicle position. Vertical green line goes through the middle of the ground image.

(a) Train, sample 458. The ground truth is at the parking space, to the left of where it should be.

(b) Train, sample 6902. The ground truth does not correspond to the turn geometry. It should be a few meters above.

(c) Train, sample 10770. The ground truth is on top of a building.

(d) Train, sample 11047. The ground truth is beyond the edge of the road.

Figure C-2: Examples of large-scale ground truth errors on the training set. Green arrow represents the ground truth vehicle position. Vertical green line goes through the middle of the ground image.

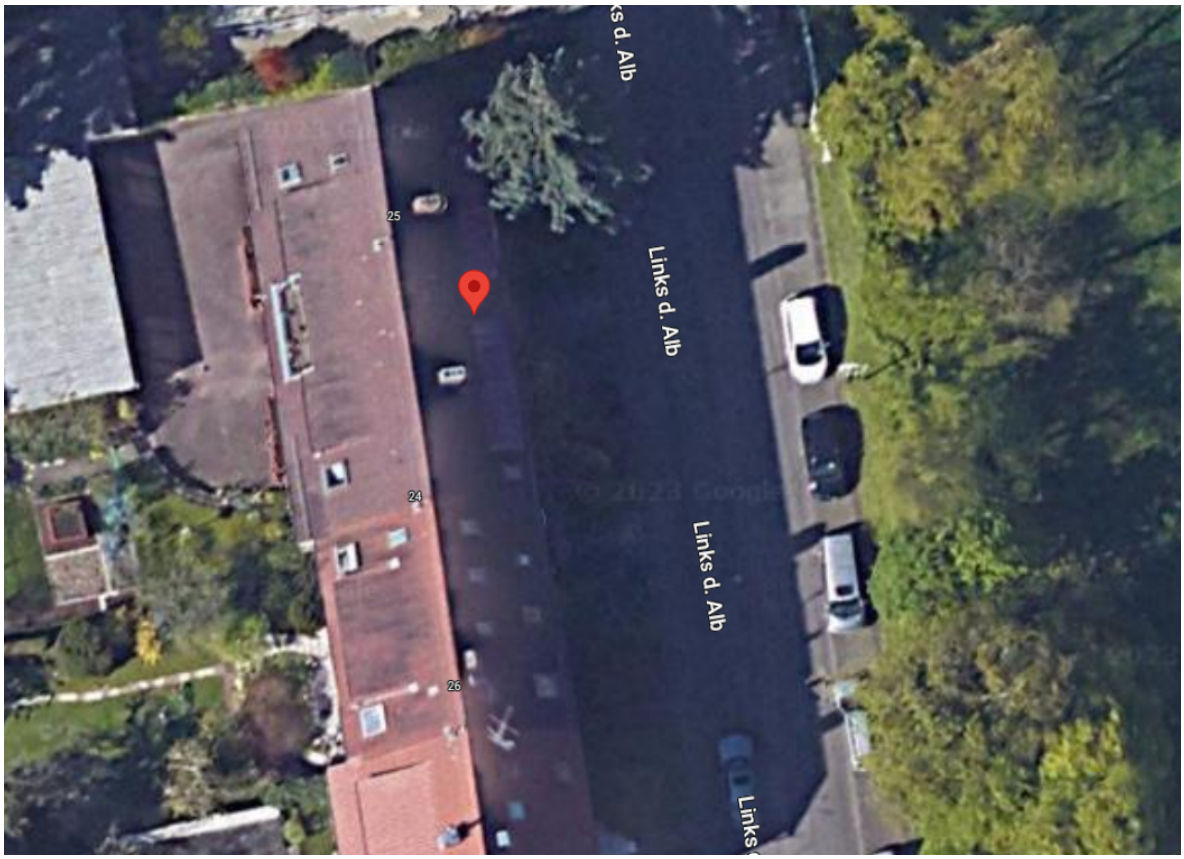


Figure C-3: GoogleMaps query of the GPS coordinates of sample 10770. The ground truth is on top of a building.

file with information about the sample contains the GPS coordinates of the ground truth as (48.983342966965, 8.3962619593469). A simple query of the coordinates using GoogleMaps shows that this point is indeed on top of a building, as shown on Figure C-3. This means that the ground truth poses in the KITTI dataset are indeed incorrect, which is a significant finding.

C-2 Small-Scale Ground Truth Errors

On the small scale, we made use of our 6-Degree of Freedom (DoF) projective transformation introduced in subsection 3-5-2. Using the 6-DoF transformation, the aerial and ground images projected onto the same domain should overlap perfectly. Note that due to the homography assumption only the ground plane is guaranteed to be transformed correctly, however we expect there to be enough keypoints to assess the quality of the ground truth.

Another important note is that the 6-DoF is really only a 5-DoF transformation, since the height of the vehicle above the ground is not present in the KITTI dataset. However, neglecting height is quite reasonable since it both has little effect even on the distant areas of the image and the height of the camera itself is not expected to change significantly during the course of the dataset.

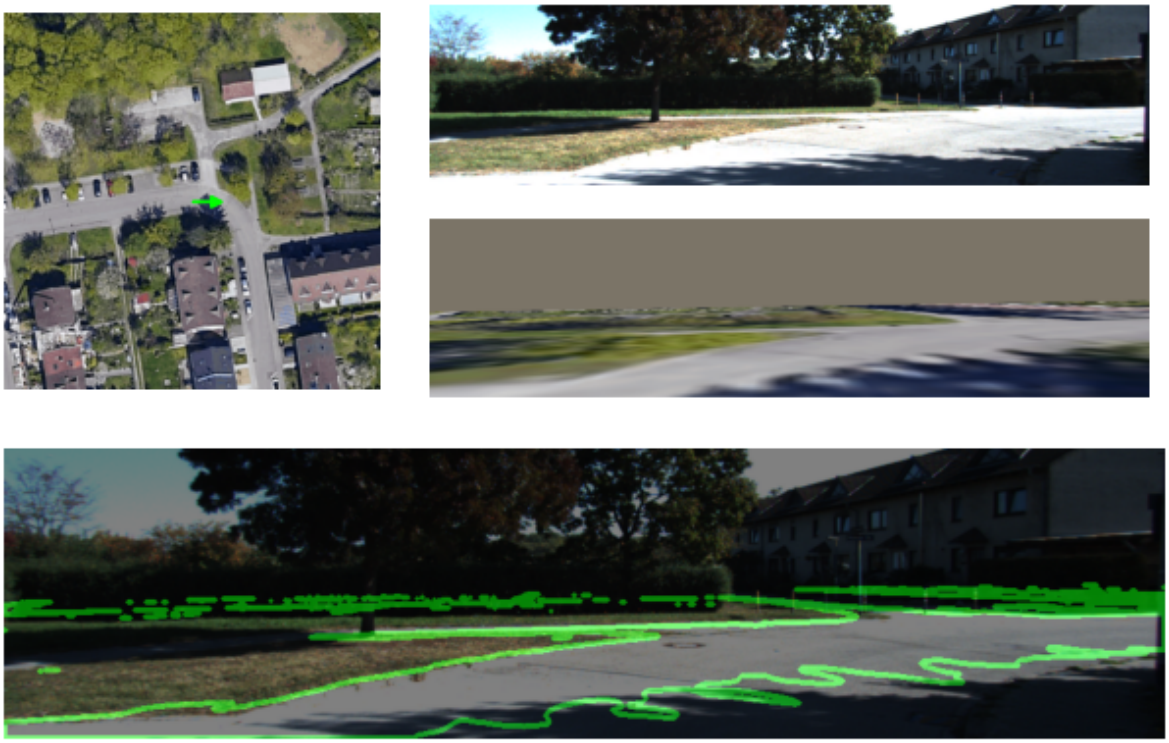
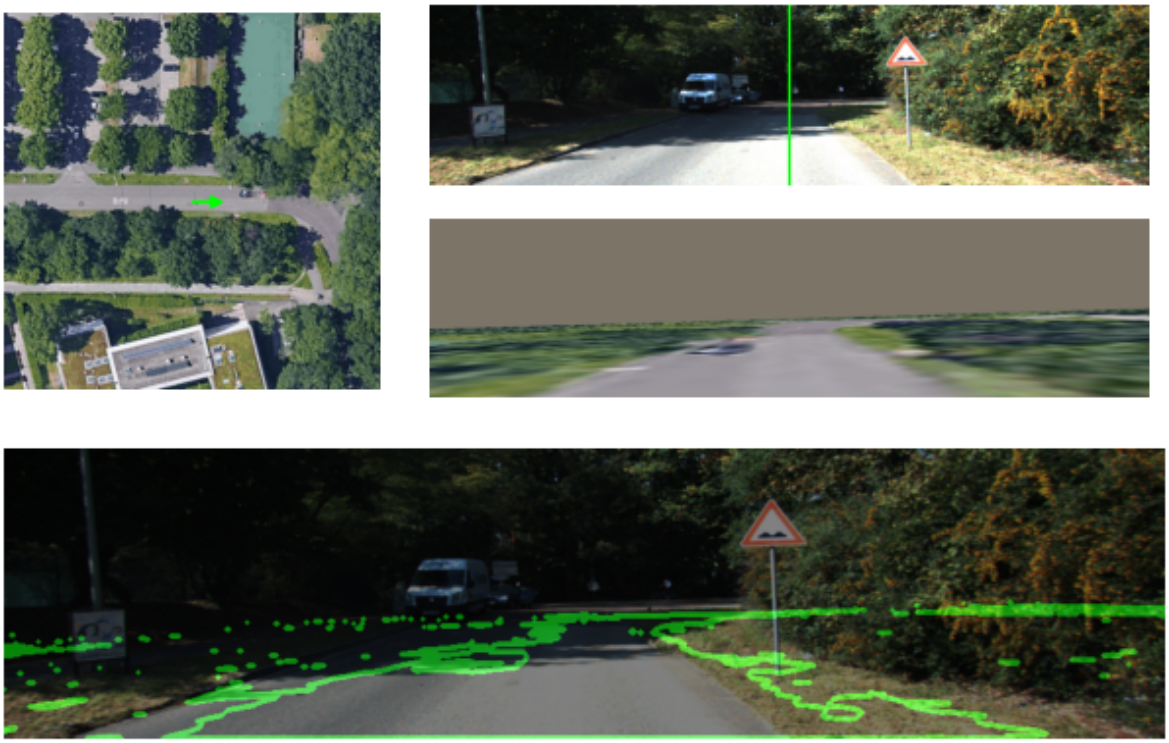


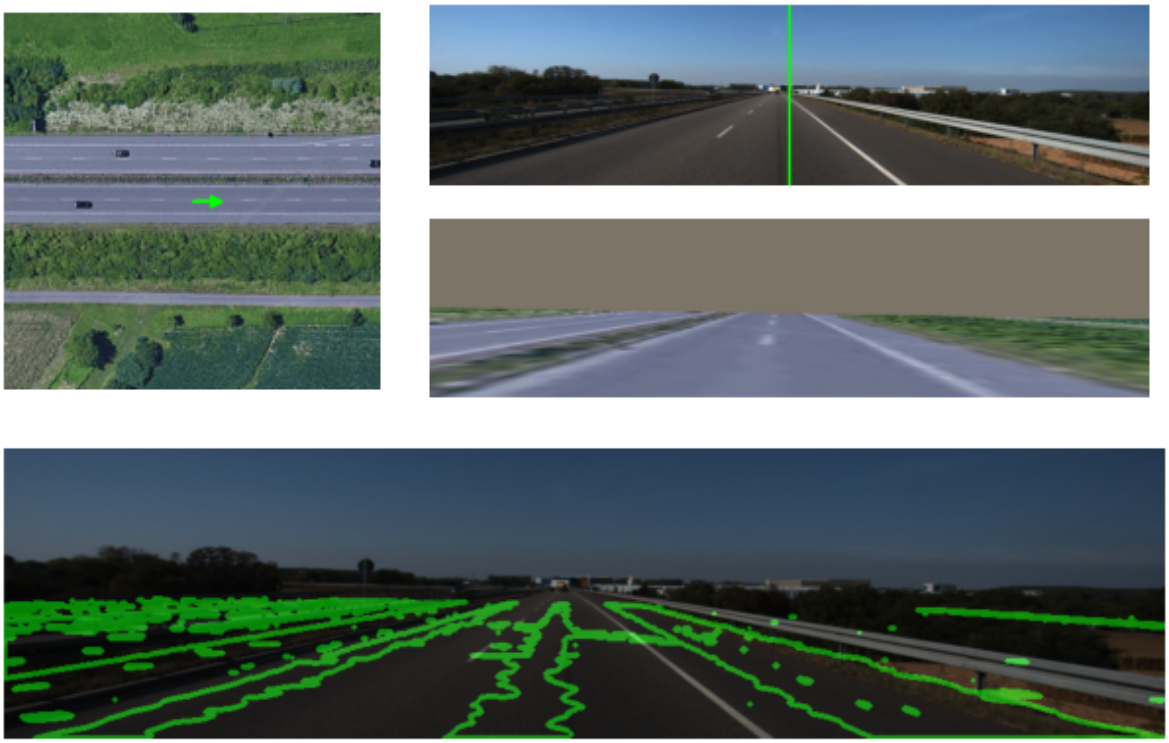
Figure C-4: Test1, sample 2709. Example of a good ground truth. Top left image: aerial image with the green ground truth arrow. Top right: ground camera feed. Middle right: transformed aerial image onto the ground domain with respect to the ground truth. Bottom: ground image overlaid with the contour map of the transformed aerial image. Note that we do not expect all lines to align perfectly, since not all contours correspond to the ground plane. Moreover, very small errors can be attributed to not accounting for the camera height changes. However, when at least a few lines align well, we can conclude that the quality of the ground truth is high.

An example of the usage of the projective transform is depicted and explained on Figure C-4. Examples of bad ground truths, where the aerial and ground images do not align well, are given on Figure C-5. The same legend as in Figure C-4 applies.

Most samples are not possible to assess due to shadows and trees, which make the projected aerial image's contours noisy. However, of those that are possible to compare, the majority have some degree of misalignment, both on the training and evaluation sets.



(a) Train, sample 295. Example of a bad ground truth.



(b) Train, sample 18949. Example of a bad ground truth.

Figure C-5: Examples of bad ground truth at the small scale. The road curbs and lane markings are not aligned well.

Bibliography

[1] K. Bimbraw, "Autonomous cars: Past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology," in *International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, vol. 1, pp. 191–198, IEEE, 2015.

[2] R. Hussain and S. Zeadally, "Autonomous cars: Research results, issues, and future challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1275–1313, 2019.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[4] J. Bressler, P. Reisdorf, M. Obst, and G. Wanielik, "Gnss positioning in non-line-of-sight context—a survey," *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016.

[5] L. N. Thin, L. Y. Ting, N. A. Husna, and M. H. Husin, "Gps systems literature: inaccuracy factors and effective solutions," *Int. J. Comput. Netw. Commun*, vol. 8, no. 2, pp. 123–131, 2016.

[6] B. Ben-Moshe, E. Elkin, H. Levi, and A. Weissman, "Improving accuracy of gnss devices in urban canyons.," in *CCCG*, pp. 511–515, 2011.

[7] Z. Xia, O. Booij, M. Manfredi, and J. F. Kooij, "Geographically local representation learning with a spatial prior for visual localization," in *Proc. of European Conference on Computer Vision (ECCV)*, pp. 557–573, Springer, 2020.

[8] Z. Xia, O. Booij, M. Manfredi, and J. F. Kooij, "Cross-view matching for vehicle localization by learning geographically local representations," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5921–5928, 2021.

[9] M. A. Al-Khedher, "Hybrid GPS-GSM localization of automobile tracking system," *CoRR*, vol. abs/1201.2630, 2012.

[10] E. Wang, W. Zhao, and M. Cai, "Research on improving accuracy of gps positioning based on particle filter," in *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, pp. 1167–1171, 2013.

[11] Z. Wu, M. Yao, H. Ma, and W. Jia, "Improving accuracy of the vehicle attitude estimation for low-cost ins/gps integration aided by the gps-measured course angle," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 553–564, 2013.

[12] S. Godha and M. Cannon, "Gps/mems ins integrated system for navigation in urban areas," *Gps Solutions*, vol. 11, no. 3, pp. 193–203, 2007.

[13] D. K. Schrader, B.-C. Min, E. T. Matson, and J. E. Dietz, "Combining multiple, inexpensive gps receivers to improve accuracy and reliability," in *2012 IEEE Sensors Applications Symposium Proceedings*, pp. 1–6, 2012.

[14] D. DeVon, T. Holzer, and S. Sarkani, "Minimizing uncertainty and improving accuracy when fusing multiple stationary gps receivers," in *2015 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 83–88, 2015.

[15] R. Ross and R. Hoque, "Augmenting gps with geolocated fiducials to improve accuracy for mobile robot applications," *Applied Sciences*, vol. 10, no. 1, 2020.

[16] L. Ge, S. Han, and C. Rizos, "Multipath mitigation of continuous gps measurements using an adaptive filter," *GPS solutions*, vol. 4, no. 2, pp. 19–30, 2000.

[17] H. G. Seif and X. Hu, "Autonomous driving in the icity—hd maps as a key challenge of the automotive industry," *Engineering*, vol. 2, no. 2, pp. 159–162, 2016.

[18] R. W. Wolcott and R. M. Eustice, "Visual localization within lidar maps for automated urban driving," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 176–183, 2014.

[19] K. Yoneda, H. Tehrani, T. Ogawa, N. Hukuyama, and S. Mita, "Lidar scan feature for localization with highly precise 3-d map," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 1345–1350, 2014.

[20] I. A. Barsan, S. Wang, A. Pokrovsky, and R. Urtasun, "Learning to localize using a lidar intensity map," *arXiv preprint arXiv:2012.10902*, 2020.

[21] J. Guo, P. V. K. Borges, C. Park, and A. Gawel, "Local descriptor for robust place recognition using lidar intensity," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1470–1477, 2019.

[22] K. Żywanowski, A. Banaszczyk, M. R. Nowicki, and J. Komorowski, "Minkloc3d-si: 3d lidar place recognition with sparse convolutions, spherical coordinates, and intensity," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1079–1086, 2022.

[23] K. Chen, L. He, X. Wang, Y. Liu, and M. Zhao, "Clmm-net: Robust cascaded lidar map matching based on multi-level intensity map," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5484–5490, 2021.

[24] J. L. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler, "Semantic visual localization," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6896–6906, 2018.

[25] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera pose voting for large-scale image-based localization," in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pp. 2704–2712, 2015.

[26] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii, "Inloc: Indoor visual localization with dense matching and view synthesis," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7199–7209, 2018.

[27] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12716–12725, 2019.

[28] T. Sattler, B. Leibe, and L. Kobbelt, "Efficient & effective prioritized matching for large-scale image-based localization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 9, pp. 1744–1756, 2016.

[29] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson, "City-scale localization for cameras with known vertical direction," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1455–1461, 2016.

[30] X. Wei, I. A. Bârsan, S. Wang, J. Martinez, and R. Urtasun, "Learning to localize through compressed binary maps," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10316–10324, 2019.

[31] Synced, "The golden age of hd mapping for autonomous driving," Aug 2018.

[32] M. Zaffar, S. Garg, M. Milford, J. Kooij, D. Flynn, K. McDonald-Maier, and S. Ehsan, "Vpr-bench: An open-source visual place recognition evaluation framework with quantifiable viewpoint and appearance change," *International Journal of Computer Vision*, vol. 129, no. 7, p. 2136–2174, 2021.

[33] Y. Shi and H. Li, "Beyond cross-view image retrieval: Highly accurate vehicle localization using satellite image," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17010–17020, 2022.

[34] T. d. V. Lentsch, Z. Xia, H. Caesar, and J. F. Kooij, "Slicematch: Geometry-guided aggregation for cross-view pose estimation," *arXiv preprint arXiv:2211.14651*, 2022.

[35] Z. Xia, O. Booij, and J. F. Kooij, "Convolutional cross-view pose estimation," *arXiv preprint arXiv:2303.05915*, 2023.

[36] O. Özyeşil, V. Voroninski, R. Basri, and A. Singer, "A survey of structure from motion*.," *Acta Numerica*, vol. 26, pp. 305–364, 2017.

[37] V. Mohanty, S. Agrawal, S. Datta, A. Ghosh, V. D. Sharma, and D. Chakravarty, "Deepvo: A deep learning approach for monocular visual odometry," *arXiv preprint arXiv:1611.06069*, 2016.

[38] Z. Xia, O. Booij, M. Manfredi, and J. F. Kooij, "Visual cross-view metric localization with dense uncertainty estimates," in *Proc. of European Conference on Computer Vision (ECCV)*, pp. 90–106, Springer, 2022.

[39] F. Fervers, S. Bullinger, C. Bodensteiner, M. Arens, and R. Stiefelhagen, "Uncertainty-aware vision-based metric cross-view geolocalization," *arXiv preprint arXiv:2211.12145*, 2022.

[40] S. International, "J3016_202104: Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," Apr 2021.

[41] C. Basnayake, T. Williams, P. Alves, and G. Lachapelle, "Can gnss drive v2x?," *GPS World*, vol. 21, no. 10, pp. 35–43, 2010.

[42] National Highway Traffic Safety Administration and Department of Transportation, *Federal Motor Vehicle Safety Standards; V2V Communications, Docket No. NHTSA-2016-0126, RIN 2127AL55*, vol. 82. National Highway Traffic Safety Administration, 8 ed., 2017.

[43] T. G. Reid, S. E. Houts, R. Cammarata, G. Mills, S. Agarwal, A. Vora, and G. Pandey, "Localization requirements for autonomous vehicles," in *Society of Automotive Engineers World Congress Experience (SAE WCX)*, SAE, 2019.

[44] D. Wilson, X. Zhang, W. Sultani, and S. Wshah, "Visual and object geo-localization: A comprehensive survey," *arXiv preprint arXiv:2112.15202*, 2021.

[45] Y. Shi, L. Liu, X. Yu, and H. Li, "Spatial-aware feature aggregation for image based cross-view geo-localization," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[46] J. Zhao, Q. Zhai, R. Huang, and H. Cheng, "Mutual generative transformer learning for cross-view geo-localization," *arXiv preprint arXiv:2203.09135*, 2022.

[47] Y. Shi, X. Yu, D. Campbell, and H. Li, "Where am i looking at? joint location and orientation estimation by cross-view matching," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4064–4072, 2020.

[48] X. Lu, S. Luo, and Y. Zhu, "It's okay to be wrong: Cross-view geo-localization with step-adaptive iterative refinement," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–13, 2022.

[49] A. Toker, Q. Zhou, M. Maximov, and L. Leal-Taixé, "Coming down to earth: Satellite-to-street view synthesis for geo-localization," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6488–6497, 2021.

[50] Y. Shi, D. Campbell, X. Yu, and H. Li, "Geometry-guided street-view panorama synthesis from satellite imagery," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 10009–10022, 2022.

[51] S. Li, Z. Tu, Y. Chen, and T. Yu, "Multi-scale attention encoder for street-to-aerial image geo-localization," *CAAI Transactions on Intelligence Technology*, 2022.

[52] K. Regmi and M. Shah, "Bridging the domain gap for ground-to-aerial image matching," in *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 470–479, 2019.

[53] S. Hu, M. Feng, R. M. Nguyen, and G. H. Lee, "Cvm-net: Cross-view matching network for image-based ground-to-aerial geo-localization," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7258–7267, 2018.

[54] L. Liu and H. Li, "Lending orientation to neural networks for cross-view geo-localization," in *Proc. of IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5624–5633, 2019.

[55] S. Cai, Y. Guo, S. Khan, J. Hu, and G. Wen, "Ground-to-aerial image geo-localization with a hard exemplar reweighting triplet loss," in *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8391–8400, 2019.

[56] Y. Shi, X. Yu, L. Liu, T. Zhang, and H. Li, "Optimal feature transport for cross-view image geo-localization," in *Proc. of AAAI Conference on Artificial Intelligence*, vol. 34, pp. 11990–11997, 2020.

[57] H. Yang, X. Lu, and Y. Zhu, "Cross-view geo-localization with layer-to-layer transformer," *Advances in Neural Information Processing Systems*, vol. 34, pp. 29009–29020, 2021.

[58] S. Zhu, M. Shah, and C. Chen, "Transgeo: Transformer is all you need for cross-view image geo-localization," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1162–1171, 2022.

[59] G. Wang, J. Chen, M. Dai, and E. Zheng, "Wamf-fpi: A weight-adaptive multi-feature fusion network for uav localization," *Remote Sensing*, vol. 15, no. 4, p. 910, 2023.

[60] X. Wu, Q. Ma, Q. Li, Y. Yu, and W. Liu, "Cross-view geo-localization based on cross-domain matching," in *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery: Proc. of ICNC-FSKD 2022*, pp. 719–728, Springer, 2023.

[61] S. Verde, T. Resek, S. Milani, and A. Rocha, "Ground-to-aerial viewpoint localization via landmark graphs matching," *IEEE Signal Processing Letters*, vol. 27, pp. 1490–1494, 2020.

[62] T. Wang, Z. Zheng, C. Yan, J. Zhang, Y. Sun, B. Zheng, and Y. Yang, "Each part matters: Local patterns facilitate cross-view geo-localization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 2, pp. 867–879, 2021.

[63] S. Zhu, T. Yang, and C. Chen, "Revisiting street-to-aerial view image geo-localization and orientation estimation," in *Proc. of IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 756–765, 2021.

[64] Y. Shi, X. Yu, D. Campbell, and H. Li, "Where am i looking at? joint location and orientation estimation by cross-view matching," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4064–4072, 2020.

[65] W. Hu, Y. Zhang, Y. Liang, Y. Yin, A. Georgescu, A. Tran, H. Kruppa, S.-K. Ng, and R. Zimmermann, "Beyond geo-localization: Fine-grained orientation of street-view images by cross-view matching with satellite imagery," in *Proc. of 30th ACM International Conference on Multimedia*, pp. 6155–6164, 2022.

[66] L. M. Downes, T. J. Steiner, R. L. Russell, and J. P. How, "Wide-area geolocalization with a limited field of view camera," *arXiv preprint arXiv:2209.11854*, 2022.

[67] R. Rodrigues and M. Tani, "Global assists local: Effective aerial representations for field of view constrained image geo-localization," in *Proc. of IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3871–3879, 2022.

[68] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen, "Twins: Revisiting the design of spatial attention in vision transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9355–9366, 2021.

[69] S. Agarwal, A. Vora, G. Pandey, W. Williams, H. Kourous, and J. McBride, "Ford multi-av seasonal dataset," *The International Journal of Robotics Research*, vol. 39, no. 12, pp. 1367–1376, 2020.

[70] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[71] T. Roddick and R. Cipolla, "Predicting semantic map representations from images using pyramid occupancy networks," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11138–11147, 2020.

[72] A. Saha, O. Mendez, C. Russell, and R. Bowden, "Enabling spatio-temporal aggregation in birds-eye-view vehicle estimation," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5133–5139, IEEE, 2021.

[73] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12697–12705, 2019.

[74] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d," in *Proc. of European Conference on Computer Vision (ECCV)*, pp. 194–210, Springer, 2020.

[75] A. Hu, Z. Murez, N. Mohan, S. Dudas, J. Hawke, V. Badrinarayanan, R. Cipolla, and A. Kendall, "Fiery: future instance prediction in bird's-eye view from surround monocular cameras," in *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15273–15282, 2021.

[76] A. Saha, O. Mendez, C. Russell, and R. Bowden, "Translating images into maps," in *International Conference on Robotics and Automation (ICRA)*, pp. 9200–9206, IEEE, 2022.

[77] P.-E. Sarlin, D. DeTone, T.-Y. Yang, A. Avetisyan, J. Straub, T. Malisiewicz, S. R. Bulo, R. Newcombe, P. Kontschieder, and V. Balntas, "Orienternet: Visual localization in 2d

public maps with neural matching," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21632–21642, 2023.

[78] L. Peng, Z. Chen, Z. Fu, P. Liang, and E. Cheng, "Bevsegformer: Bird's eye view semantic segmentation from arbitrary camera rigs," in *Proc. of IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 5935–5943, 2023.

[79] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai, "Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *Proc. of European Conference on Computer Vision (ECCV)*, pp. 1–18, Springer, 2022.

[80] Y. Liu, T. Wang, X. Zhang, and J. Sun, "Petr: Position embedding transformation for multi-view 3d object detection," in *Proc. of European Conference on Computer Vision (ECCV)*, pp. 531–548, Springer, 2022.

[81] S. Zhu, T. Yang, and C. Chen, "Vigor: Cross-view image geo-localization beyond one-to-one retrieval," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3640–3649, 2021.

[82] Y. Shi, X. Yu, L. Liu, D. Campbell, P. Koniusz, and H. Li, "Accurate 3-dof camera geo-localization via ground-to-satellite image matching," *arXiv preprint arXiv:2203.14148*, 2022.

[83] S. Workman, R. Souvenir, and N. Jacobs, "Wide-area image geolocalization with aerial reference imagery," in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pp. 3961–3969, 2015.

[84] L. Liu and H. Li, "Lending orientation to neural networks for cross-view geo-localization," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5624–5633, 2019.

[85] Y. Hou, Y. Yang, J. Wang, and M. Fu, "Road extraction assisted offset regression method in cross-view image-based geo-localization," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2934–2940, IEEE, 2022.

[86] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[87] W. Maddern, G. Pascoe, M. Gadd, D. Barnes, B. Yeomans, and P. Newman, "Real-time kinematic ground truth for the oxford robotcar dataset," *arXiv preprint arXiv:2002.10152*, 2020.

[88] Y. Liao, J. Xie, and A. Geiger, "Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[89] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8748–8757, 2019.

[90] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, *et al.*, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," *arXiv preprint arXiv:2301.00493*, 2023.

[91] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska, "One thousand and one hours: Self-driving motion prediction dataset," in *Conference on Robot Learning*, pp. 409–418, PMLR, 2021.

[92] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11621–11631, 2020.

[93] P. Xiao, Z. Shao, S. Hao, Z. Zhang, X. Chai, J. Jiao, Z. Li, J. Wu, K. Sun, K. Jiang, *et al.*, "Pandaset: Advanced sensor suite dataset for autonomous driving," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 3095–3101, IEEE, 2021.

[94] S. Wang, Y. Zhang, and H. Li, "Satellite image based cross-view localization for autonomous vehicle," *arXiv preprint arXiv:2207.13506*, 2022.

[95] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[96] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[97] F. Barbato, G. Rizzoli, and P. Zanuttigh, "Depthformer: Multimodal positional encodings and cross-input attention for transformer-based segmentation networks," *arXiv preprint arXiv:2211.04188*, 2022.

[98] F. I. Diakogiannis, F. Waldner, P. Caccetta, and C. Wu, "Resunet-a: A deep learning framework for semantic segmentation of remotely sensed data," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 162, pp. 94–114, 2020.

[99] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. of IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Ieee, 2009.

[100] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[101] D. Chen, J. Li, V. Guizilini, R. A. Ambrus, and A. Gaidon, "Viewpoint equivariance for multi-view 3d object detection," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9213–9222, 2023.

[102] Y. Cui, L. Yang, and H. Yu, "Dq-det: Learning dynamic query combinations for transformer-based object detection and segmentation," *arXiv preprint arXiv:2307.12239*, 2023.

[103] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. of European Conference on Computer Vision (ECCV)*, pp. 213–229, Springer, 2020.

[104] S. Wang, Y. Zhang, A. Perincherry, A. Vora, and H. Li, "View consistent purification for accurate cross-view localization," *arXiv preprint arXiv:2308.08110*, 2023.

[105] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11976–11986, 2022.

[106] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *International journal of computer vision*, vol. 128, pp. 261–318, 2020.

[107] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.

[108] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12077–12090, 2021.

[109] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[110] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Glossary

List of Acronyms

A2G	Aerial to Ground
BEV	Bird's-Eye View
CNN	Convolutional Neural Network
CVGL	Cross-View Geo-Localization
CVIR	Cross-View Image Retrieval
CVM	Cross-View Matching
CVPE	Cross-View Pose Estimation
DoF	Degree of Freedom
FoV	Field of View
GAN	Generative Adversarial Network
G2A	Ground to Aerial
GICP	Generalized Iterative Closest Point
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HD	High Definition
ICP	Iterative Closest Point
LM	Levenberg-Marquardt
LiDAR	Light Detection and Ranging
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NHTSA	National Highway Traffic Safety Administration
NLL	Negative Log-Likelihood
P2P	Point-to-Point
PCA	Principal Component Analysis

R2P Ray-to-Point
RTK Real-Time Kinematic
SAE Society of Automotive Engineers
SfM Structure from Motion
TRF Trust Region Reflective
UAV Unmanned Aerial Vehicle
V2V Vehicle-to-Vehicle
V2X Vehicle-to-Everything
VGG Visual Geometry Group
VIO Visual-Inertial Odometry
ViT Vision Transformer
VO Visual Odometry
VPR Visual Place Recognition