SliceNet: Street-to-Satellite Image Metric Localization using Local Feature Matching

T. de Vries Lentsch

Supervisors J.F.P. Kooij and Z. Xia

Tuesday 4th October, 2022

TU Delft

SliceNet: Street-to-Satellite Image Metric Localization using Local Feature Matching

by

T. de Vries Lentsch

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday October 11, 2022 at 10.45 AM.

Student number:	4668804	
Project duration:	November 17, 2021 – October 4, 2022	
Thesis committee:	Dr. J.F.P. Kooij,	TU Delft, supervisor and committee chair
	Ir. Z. Xia,	TU Delft, daily supervisor and committee member
	Dr. H. Caesar,	TU Delft, Cognitive Robotics committee member
	Dr. S. Khademi,	TU Delft, external committee member

TU Delft

Abstract

This work addresses visual localization for intelligent vehicles. The task of cross-view matching-based localization is to estimate the geo-location of a vehicle-mounted camera by matching the captured street view image with an overhead-view satellite map containing the vehicle's local surroundings. This local satellite view image can be obtained using any rough localization prior, e.g., from a global navigation satellite system or temporal filtering. Existing cross-view matching methods are global image descriptor-based and achieve considerably lower localization performance than structure-based methods with 3D maps. Whereas structure-based methods utilized global image descriptors in the past, recent structure-based work has shown that significantly better localization performance can be achieved using local image descriptors to find pixel-level correspondences between the query street view image and the 3D map. Hence, using local image descriptors may be the key to improving the localization performance of cross-view matching methods. However, the street and the satellite view do exhibit not only very different visual appearances but also have distinctive geometric configurations. As a result, finding correspondences between the two views is not a trivial task. We observe that the geometric relationship between the street and satellite view implies that every vertical line in the street view image has a corresponding azimuth direction in the satellite view image. Based on this prior, we devise a novel neural network architecture called SliceNet that extracts local image descriptors from both images and matches these to compute a dense spatial distribution for the camera's location. Specifically, the geometric prior is used as a weakly supervised signal to enable SliceNet to learn the correspondences between the two views. As an additional task, we also show that the extracted local image descriptors can be used to determine the heading of the camera. SliceNet outperforms global image descriptor-based cross-view matching methods and achieves state-of-the-art localization results on the VIGOR dataset. Notably, the proposed method reduces the median metric localization error by 21% and 4% compared to the state-of-the-art methods when generalizing, respectively, in the same area and across areas.

Contents

List of Figures

List of Tables

1 Introduction

According to the World Health Organization [53], about 1.3 million people die each year due to road traffic crashes. Road traffic injuries are even the leading cause of death for children and young adults aged 5-29 years. Moreover, road traffic injuries cause considerable economic losses to individuals, their families, and nations. Since care is needed for the injured persons, the productivity of these persons may decrease, and family members may not be able to continue going to school or work. Therefore, in 2021, the United Nations General Assembly has set an ambitious target of halving the global number of deaths and injuries from road traffic crashes by 2030 [53].

Autonomous vehicles could play a significant role in assuring road safety [20] as more than 90 percent of all accidents are caused by human error [19]. Besides, automated driving could improve traffic flows, promote the mobility and productivity of people, and improve driving comfort [96]. Highly accurate and detailed lane-level maps, also known as *high-definition maps*, are crucial for safe automated driving [97]. They contain precise definitions of the road lanes, augment vehicle sensor information for contextual analysis of the environment, and assist the vehicle in executing controlled maneuvers beyond its sensing range [98]. In order to use these maps, an accurate localization technique is required to provide vehicle positioning in map coordinates.

Vehicle localization is the task of determining a vehicle's location relative to a given map of the environment [78]. The high-definition map is defined in a coordinate system of the environment, which is different from the vehicle's local coordinate system. Global localization can establish the correspondence between these two coordinate systems, but the coordinate transformation cannot be measured directly with a sensor [78].

A global navigation satellite system, i.e., GNSS, is commonly used to estimate the vehicle's location as it can provide global positioning without error accumulation [8, 11]. However, GNSS is only effective in places with a clear sky view because GNSS receivers tend to output inaccurate location estimations when operating in urban regions [2, 8]. This is primarily due to tall buildings, which often block a receiver's line of sight to the navigation satellites. Modern GNSS receivers are sensitive enough to receive the indirect signal reflected from the buildings, but this can still result in poor performance in urban canyons with errors of tens of meters [8, 90, 91]. These errors are in strong contrast with the requirements for automated driving systems based on the road geometry standards of the United States [58], namely, for vehicles operating on freeway roads, the required lateral en longitudinal error upper bounds are 0.57 and 1.40 meters, respectively. The road geometry makes requirements even more stringent on local streets with lateral and longitudinal error upper bounds of 0.29 meters.

Suppose the situation where an autonomous vehicle drives through an urban region (see Fig. 1.1). The vehicle has multiple onboard sensors, such as a GNSS receiver and an RGB camera. The GNSS receiver can provide a coarse localization, but this signal has an offset of several meters. Therefore it is not known in which lane the vehicle is driving. However, sensing the environment can be used to improve the localization [30, 62, 72, 92, 107]. The vehicle's local surroundings are visible to the camera, and thus the camera images can be used to determine a more accurate localization using *visual localization*. The vehicle's localization process then consists of two steps. The first step is to get a coarse localization for the vehicle using any rough localization prior, e.g., wheel odometry (see green arrow in Fig. 1.1), GNSS (see orange arrow in Fig. 1.1), or temporal filtering. The second step is to use visual localization to improve the vehicle's localization and get a fine localization (see yellow arrow

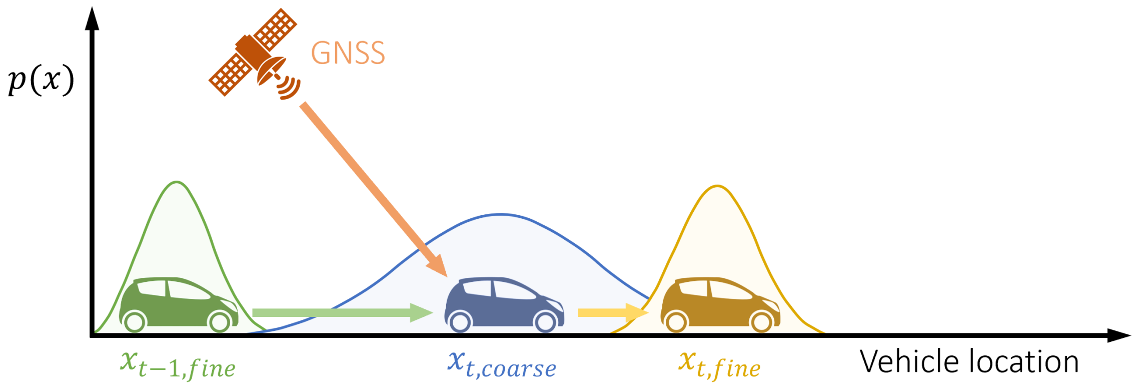


Figure 1.1: The localization of an autonomous vehicle. A coarse localization can be obtained using any rough localization prior, e.g., from wheel odometry in combination with an old location, a GNSS, or temporal filtering. The green and the orange arrow indicate wheel odometry and GNSS-based localization, respectively. Both techniques can approximate the vehicle location at time t. A more accurate localization technique, e.g., visual localization, can be used to refine the localization using the coarse vehicle location (yellow arrow). The function $p(x)$ indicates the probability density at vehicle location x.

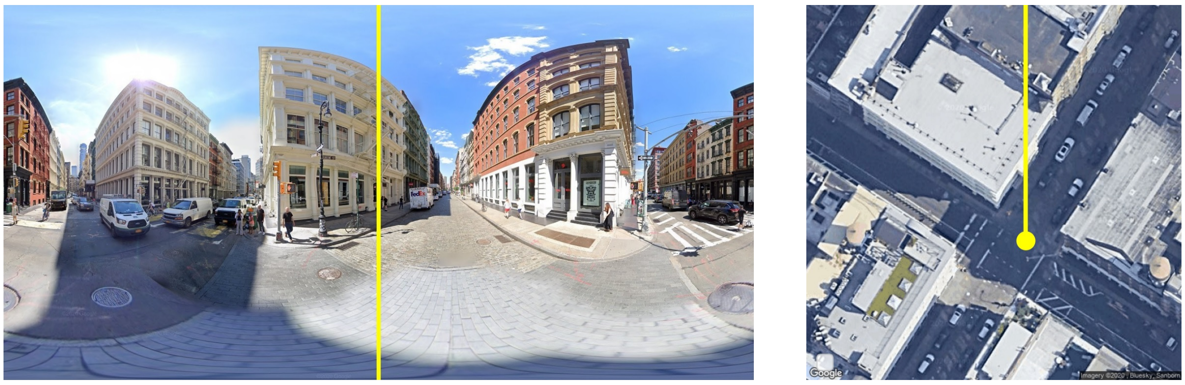
in Fig. 1.1). The advantage of cameras to sense the environment over other onboard sensors such as Radar and LiDAR is that cameras are relatively cheap and can provide significantly more semantic information. Besides, they have the desirable advantages of detecting long-range distance objects and identifying vision-based road elements [38].

Visual localization [3, 15, 16, 50, 62, 64] has attracted considerable attention in the past few years because of its practical applications in various fields, including autonomous driving, virtual and augmented reality, and digital forensics. The technique aims to determine the geo-location of a query image using a reference database of geo-tagged images. Usually, visual localization uses a street view image for both the query and the reference images. However, the disadvantage of using street images as references is that they are not available for all areas, and it takes a lot of effort to keep such a database up to date. In contrast, satellite databases are available that densely cover the Earth and the collection of new satellite images is much more convenient. As a result, matching street images to satellite images has become a popular approach for visual localization [29, 72, 73, 85, 88, 92, 107].

Street-to-satellite image matching, also known as *cross-view matching*, is challenging because of the drastic viewpoint difference between the street and the satellite view. The viewpoint difference causes that typically the side of an object is visible in the street image, while the satellite image shows the top of the object. For example, in Fig. 1.2, it can be seen that the street image shows the facades of some buildings, while the satellite image shows roofs. Even for a human, it can be challenging to figure out which roof from the satellite image belongs to which house from the street image. In addition, perspective is only present in the street view. The size with which an object appears in the street images depends on the object's distance to the street camera. In contrast, in the satellite image, two equal-sized objects appear equal-sized because of the relatively large distance between the ground and the satellite. Apart from the viewpoint difference, there is also a difference in capture time. In practice, the satellite images are captured earlier than the street images as the satellite images are needed for the reference database. As a result, different weather and light conditions may occur, which can cause the appearance of both images to differ significantly. Besides, the difference in capture time can also affect the presence of non-stationary objects.

Early cross-view matching work formulated the localization task as an instance of *image retrieval* [29, 73, 85, 88]. For each query street image, the most similar satellite image is searched from the reference database, and the geo-tag of the retrieved image is used as a rough estimate of the location of the street camera. However, a rough estimate can also be obtained with GNSS [92]. Moreover, the localization accuracy is limited by the sample density of database images, so there is a trade-off between computational efficiency and localization accuracy [72].

Recent cross-view matching work [72, 82, 92, 107] makes the switch from image retrieval to *image metric localization*, i.e., meter-level localization, and proposes to localize the street camera in a single satellite image to obtain a more accurate localization. Some methods [72, 92] also determine the orientation of the street camera so that a 3-DoF pose is predicted. The satellite image showing the vehicle's local surroundings can be obtained using any rough localization prior (see Fig. 1.1).



(a) Street (360° panorama) image **(b)** Satellite image

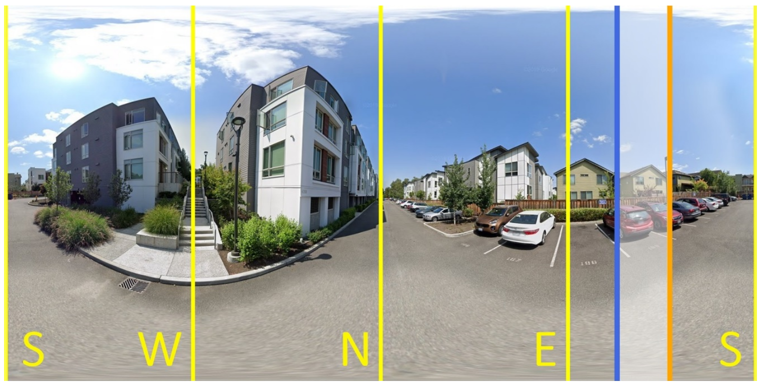
Figure 1.2: A street and satellite image pair. The yellow dot indicates the street camera location in the satellite image, and the yellow line indicates North in each image. The image pair is part of the VIGOR dataset [107].

Existing cross-view image metric localization methods [92, 107] create a global street image descriptor, i.e., a high-dimensional vector, that encodes the entire street image and one or multiple descriptors for the satellite image. The location is predicted using these descriptors, and the methods have localization errors in the order of several meters. In contrast, structure-based methods match query street images with 3D maps of the environment for estimating the 6-DoF street camera pose, and these methods achieve centimeter-level accurate localization [62, 65, 77]. Whereas structure-based methods utilized global image descriptors in the past, recent structure-based work has shown that significantly better localization performance can be obtained using local image descriptors rather than global image descriptors [62]. The local descriptors are used to get one-to-one correspondences between the query image and the 3D map. Subsequently, geometric verification is used to estimate the 6-DoF camera pose. Hence, if we can do something similar for cross-view matching, this may be the key to improving the localization performance of cross-view matching methods. Unfortunately, the viewpoint difference for cross-view image matching is considerably more significant than for structure-based matching, making matching local descriptors between the street and the satellite view more difficult. Local can mean different things depending on the context, e.g., pixel-level or object-level. In this work, we define *local* as a term that refers to object-level unless otherwise stated. Thus local descriptors can be used to obtain object-level matches between images.

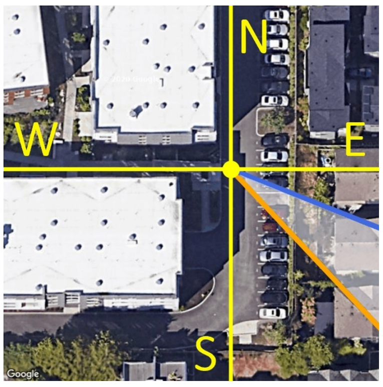
Predicting the 6-DoF camera pose using 3D maps differs from cross-view matching in several ways. First, structure-based methods use the street view for both the query and reference images, while cross-view matching uses the satellite view for the reference images. Consequently, structure-based methods can use the same feature extractor for the query and references, whereas cross-view matching requires a separate feature extractor for each view because the two views are so different [29, 62]. Second, for structure-based methods, one-to-one correspondences between a query and the reference image are available during training [62, 65, 77]. In contrast, for cross-view matching, explicitly labeled correspondences between the two views are unavailable [62, 72, 107]. However, learning local descriptors requires correspondence between the two views. Therefore, an alternative way to link local information between the two views is needed.

The geometric relationship between the street and the satellite view, i.e., the *azimuth prior*, has been noted by many cross-view matching works [37, 42, 73, 74, 75, 86, 106]. The relationship implies that every vertical line in the street image has a corresponding azimuth direction in the satellite image. Methods have been proposed that use the prior to bridge the gap between the two image domains by transforming an image from one image domain to the other image domain [37, 73, 75] and by adding the orientation information as an extra network input [42]. However, no work has used the azimuth prior for learning explicit local correspondences between the two views.

The ground truth street camera pose for each street and satellite image pair is available during training. Therefore, line pairs between the two views can be made by exploiting the prior (see the blue line in Fig. 1.3). Using two different azimuth directions (see the blue and orange line in Fig. 1.3), an image region, i.e., slice, can be delineated in both images. Two belonging slices have a visual overlap.



(a) Street (360° panorama) image



(b) Satellite image

Figure 1.3: A street and satellite image pair that shows the azimuth prior, i.e., the geometric relationship that every vertical line in the street image has a corresponding azimuth direction in the satellite image. The yellow dot indicates the street camera location in the satellite image, and the yellow lines indicate the compass strokes. Both the blue and the orange lines show an azimuth direction. These two lines delineate a slice in both images, i.e., the transparent region. The street image slice and the satellite image slice belong to each other and have a visual overlap. The image pair is part of the VIGOR dataset [107].

Local image features for cross-view matching may be learned by exploiting the azimuth prior. A street and satellite slice pair can indicate what a particular object looks like in the two views due to the visual overlap (see Fig. 1.3). However, a slice may contain multiple objects, and the visual overlap may be minimal, e.g., because an object blocks the view. Therefore, how this visual overlap can be used to learn correspondences between the two views is an open problem.

In addition, some objects do not help determine the street camera location because they are only visible in one of the two images or are not discriminative. Therefore, we also need to develop a method to select features so that sparse features can be extracted from the images. In this work, we define *sparse features* as a term that refers to a subset of a set of features.

When local correspondences between the two views can be learned, this needs to be used to determine the street camera location in the satellite image. However, an extracted feature may match several features of the other image. Therefore, an open problem is how the local image features can be matched and how the matching can be used to determine the street camera location. Based on the identified open problems, the following main research question is formulated:

How can we end-to-end learn sparse local image features across the street and the satellite view without using explicitly labeled correspondences, and how can this be used for localizing the street camera in the satellite image?

In order to answer the main research question, three subquestions are researched:

1. How can we end-to-end learn local image features across the street and satellite view without using explicitly labeled correspondences?
2. How can we extract sparse features from the street and satellite view?
3. How can we use local image feature matching for localizing the street camera in the satellite image?

Departing from the use of global image descriptors for cross-view metric localization, we devise a novel neural network architecture called SliceNet that exploits the azimuth prior in order to match sparse local image features between the street and the satellite view and compute a dense spatial distribution for the location of the street camera. Specifically, SliceNet extracts deep features from the images and uses attention to select discriminative features for both views. Subsequently, it divides both images into slices and computes a slice descriptor for each slice. Finally, SliceNet matches the slice descriptors to obtain the dense spatial distribution for the location of the street camera. We evaluate SliceNet on the VIGOR dataset [107].

Chapter structure: In Chapter 2, the literature related to street-to-satellite image matching, local image feature matching, and exploiting the azimuth prior is reviewed. Subsequently, in Chapter 3, the methodology used to answer the research questions is explained. Then, the experimental results are presented in Chapter 4. Lastly, Chapter 5 contains the conclusion along with proposed future work.

2 Related Work

In this chapter, we review the work that is the most related to street-to-satellite image matching, local image feature matching, and exploiting the geometric relationship between the street and the satellite view.

2.1. Street-to-Satellite Image Retrieval

The objective of street-to-satellite image retrieval is to find matching images from a reference database with satellite images so that the GNSS data from the retrieved images can be used as a geo-localization estimate for the query street image [29, 73, 79, 107]. The reference database is obtained by dividing a satellite map into satellite patches. The key problem is to build viewpoint and appearance invariant image representations that can be matched so that the corresponding satellite images can be retrieved for the query street image.

In general, street-to-satellite image retrieval is done by creating a global image descriptor for the query street image and each satellite image from the reference database [29, 57, 72, 73, 79, 92, 93, 106, 107]. Such a descriptor is a high-dimensional vector containing an image's encoding. The term global image descriptor is used because the descriptor is an encoding for the entire image. The concept is that the descriptor of a street image is similar to the descriptors of the matching satellite images while differing from the descriptors of non-matching satellite images. In other words, satellite images that show very different environments should have different encoding. After a global image descriptor is created for the query street image and the satellite images, the reference database can be sorted based on the similarities between the street descriptor and the satellite descriptors. The top k ranked satellite images are the output of the image retrieval system.

2.1.1. Handcrafted Descriptors

In the early stages of the street-to-satellite image retrieval field, handcrafted feature representation approaches were used to create the descriptors. These approaches rely on handcrafted feature extractors to extract representative features from the street and satellite images. Typically, the extracted features can be directly used as a global image descriptor. Features extractors such as SIFT [43] and SURF [6] were used to create a descriptor [4, 40, 49, 68, 69, 84]. These methods match by looking for object features that appear in both images. However, they failed to achieve high performance because of the difference in viewpoint between the two views [29].

2.1.2. Early Deep Learning-based Descriptors

In 2015, Workman and Jacobs [88] were the first to use a convolutional neural network, i.e., CNN, for street-to-satellite image retrieval. They used the intermediate feature representation of one of the top fully connected layers of AlexNet [36] as the global image descriptor, achieving better performance than the state-of-the-art with handcrafted features. Lin et al. [41] were the first to present a *Siamese-like network* [12, 17], i.e., a network with two branches, for street-to-oblique aerial image retrieval. These two views have more common object features because in oblique aerial images, for example, the facades of buildings can be visible. This makes the matching for street-to-oblique aerial image matching easier than for street-to-satellite image matching. Later in 2015, Workman et al. [89] proposed the MCVPlaces model, the first Siamese-like network for street-to-satellite image retrieval.

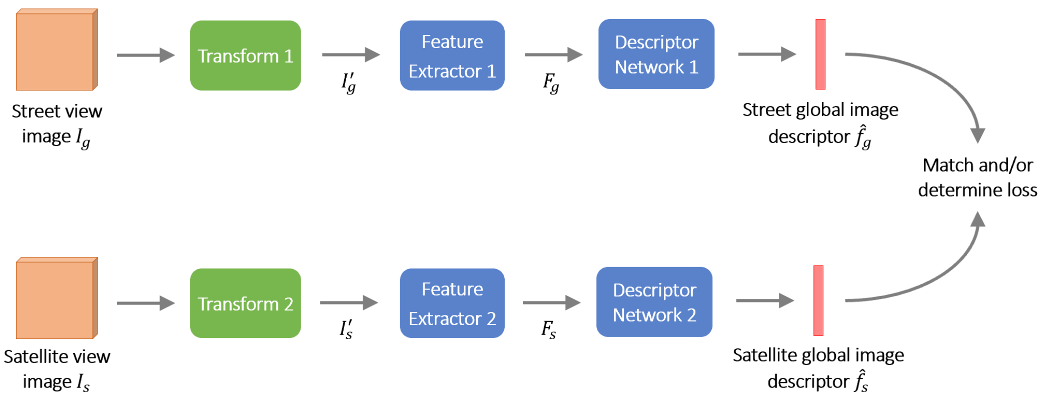


Figure 2.1: The standard street-to-satellite image retrieval framework for deep learning-based descriptors. The network has a Siamese-like architecture, containing a street and a satellite branch. The variables I, I', F, and $\hat{f}$ indicate the image, transformed image, feature map, and normalized global image descriptor of a branch, respectively. The *ground-level* street view and the satellite view are indicated by *g* and *s*, respectively, to avoid confusion.

2.1.3. Standard Framework for Deep Learning-based Descriptors

The year 2015 can be seen as the year in which the street-to-satellite image retrieval field made the switch from handcrafted descriptors methods to deep learning-based descriptor methods using Siamese-like networks [41, 88, 89]. Fig. 2.1 shows the standard framework on which post-2015 methods are typically based [29, 57, 73, 79, 90, 91, 93, 106, 107]. The network has a street and a satellite branch. Both branches consist of three components: (1) an image transformation block, (2) a feature extractor, and (3) a descriptor network. The network computes a global image descriptor for both the street and satellite image. The descriptors are used to retrieve the matching satellite images and to determine the loss. Usually, the similarity between descriptors is measured with the Euclidean distance [21, 31, 87] and the triplet loss [29, 57, 73, 79, 93, 106, 107] is used to learn discriminative descriptors.

Image transformations (see green blocks in Fig. 2.1) can be applied to reduce the image domain gap between the two images. An example is the polar transformation [73] which maps the satellite image to a synthetic street image. However, not all methods use image transformation. Therefore, this block is considered optional.

The feature extractor is usually a CNN responsible for encoding the features of a branch's (transformed) input image. The output of the CNN is a feature map and contains semantic and geometric information. Subsequently, this information is used by the descriptor network to create a global image descriptor. This way, a street and satellite descriptor are created for the street and satellite image, respectively. The descriptors are normalized so that all descriptors lie on the surface of the same high-dimensional hypersphere.

2.1.4. Deep Learning-based Descriptors

Hu et al. [29] proposed CVM-Net for learning viewpoint-invariant image descriptors. CVM-Net is a Siamese-like network using two branches, each consisting of a VGG16 [76] combined with a NetVLAD layer [3]. Cai et al. [14] proposed an attention module that uses feature and spatial attention to reweigh both the street feature map and the satellite feature map to emphasize visually salient features.

Liu and Li [42] noticed that previous methods focused on capturing image similarity in terms of visual appearance and semantic content and overlooked a crucial geometric cue for localization, namely the geometric relation between the two views. Therefore, they proposed a method that explicitly encodes the orientation of each image pixel. Shi et al. [73] observed similarly to Liu and Li. that image content visible on the pixels lying in the same azimuth direction of a satellite image approximately corresponds to the content of a column of pixels of the street image. Based on this, they proposed to use a polar transformation to map a satellite image to a synthetic street image which significantly increased the performance of their method. The polar transform is a simple approximation for the cross-view image transformation and gives a lot of visual distortion. Therefore, Shi et al. [75] proposed the projective transformation as a replacement for the polar transformation.

In contrast, Li et al. [37] tried to bridge the domain gap between the two views by using an inverse polar transformation to make the street image approximately aligned with the satellite image. The advantage of the inverse polar transformation over the polar transformation is that no street camera location needs to be assumed in the satellite image. The disadvantage, however, is that there are large visual differences between the satellite image and the synthetic satellite image.

Regmi and Shah [57] and Toker et al. [79] proposed using a generative adversarial network, i.e., GAN, to minimize the domain gap between the two views. The method of Regmi and Shah creates a synthetic satellite image and extracts features from that image. Subsequently, the extracted features are fused with the original street image's features to obtain a robust query representation. In contrast, Toker et al. create a synthetic street image instead of a synthetic satellite image. In addition, they use the features of the latent space of the GAN, while Regmi and Shah use the synthetic street image as input to a feature extractor.

Yang et al. [93] stated that existing work imposes a strong assumption on the positional knowledge by concatenating the spherical directions of each pixel to the images [42] and by using the polar transformation [73, 74]. Therefore, they proposed a method that exploits the positional encoding of the Transformer [81] to help the model to understand and correspond geometric configurations between the street and satellite view. Instead of hardcoded orientation information or a predefined transformation, they let the model learn the positional embeddings through the training objective. Besides, work has been proposed that tries to use the mutual benefits between the two views by devising two symmetrical generative sub-modules to improve the attention mechanism of the Transformer [103]. Lastly, Zhu et al. [105] adopted the Vision Transformer for cross-view image retrieval.

2.1.5. Shortcomings for Localization

During the past few years, the performance of street-to-satellite image retrieval methods has advanced significantly. Methods have been proposed to create viewpoint and appearance invariant global image descriptors. However, it is generally assumed that the street camera is located in the center of the satellite image [42, 99], while this cannot be guaranteed during inference [107]. In addition, there is a trade-off between the localization accuracy and the computational efficiency [72, 92]. To achieve meter-level accuracy, the reference satellite images must have a high overlap with each other. Thus, the global image descriptor of the query street image must be compared with many satellite descriptors, which is not computationally efficient. Therefore, it is not practical to use street-to-satellite image retrieval to geo-localize an autonomous vehicle.

2.2. Street-to-Satellite Image Metric Localization

Recent cross-view matching work [72, 82, 92, 107] goes a step further than retrieving matching satellite images and attempts to determine the location and sometimes the orientation of the street camera in the matching satellite image. This allows for obtaining a more accurate location and orientation. Moreover, this solves the localization accuracy and computational efficiency trade-off of image retrieval because only one satellite image is used.

2.2.1. Methods

The work of Verde et al. [82] can be seen as the first work that presented a street-to-satellite image metric localization method. They proposed a landmark graph matching-based method that attempts to localize a street panorama image in a satellite image without requiring information about the orientation of both images. A landmark graph is a relational graph structure that models the adjacency of salient objects in the scene [82]. The proposed method is an unsupervised approach, and it is based on the observation that proximity relationships between nearby objects or landmarks are maintained in both views. The authors manually annotated landmarks on sample image pairs to demonstrate their method. However, they did not explain how landmarks can be detected without having labeled landmarks to train a detector and left that as future work. Therefore, Verde et al.'s method is not considered to be an end-to-end method for street-to-satellite image metric localization.

Zhu et al. [107] proposed the *cross-view regression* method, which can localize a street image in a satellite image in a coarse-to-fine manner. First, image retrieval is used to obtain the matching satellite image. After that, the location of the street image is predicted in the retrieved satellite image. Location prediction is performed by concatenating the global image descriptors of both views and giving them as an input to a multilayer perceptron, i.e., MLP, that predicts a 2D image location. The network is a direct extension of the image retrieval network of Shi et al. [73]. The MLP matches the content of the two global image descriptors implicitly for predicting the street camera location. While Zhu et al. note the geometric relationship between the two views, they do not use this prior to improve the matching between them.

Xia et al. [92] proposed a classification-based method for street-to-satellite image metric localization. Rather than formulating it as a regression task as Zhu et al. did, they proposed to produce a dense multi-modal distribution to capture localization ambiguities and avoid regressing to the center between multiple visually similar places. They noted that the cross-view regression method does global coarse and local metric localization with a global image descriptor for both views. A global image descriptor may contain information about the entire image of a view, so it could be the case that the location prediction branch lacks fine-grained scene information. Therefore, Xia et al. proposed to split the satellite feature map into a grid and create a separate descriptor for each grid cell. Splitting the satellite feature map allows the method to match the content of the street global image descriptor with local regions of the satellite image. However, Xia et al. do not divide the street image into local regions. In contrast, local regions of the street image could be matched with the local regions of the satellite image to obtain one-to-one correspondences. This would also offer the possibility of using the geometric relationship between the two views to avoid certain matches and to help the network learn the correspondences.

Shi et al. [72] argued that the significant domain differences between the street and satellite view make it difficult to obtain accurate regression with the cross-view regression method. Therefore, they proposed to formulate the problem as a 3-DoF pose estimation problem and solve it using neural network-based optimization. In contrast to the methods of Zhu et al. and Xia et al., their method can do location and orientation estimation directly. Shi et al. noted that the street and satellite image overlap mainly lies on the ground plane. Therefore, they assumed that everything visible in the satellite image lies on the ground plane of the street view, i.e., at the same height as the road. A geometry projection module is used to project the satellite features on this ground plane using the camera parameters and the height of the street camera above the road. This provides a prediction of what the street view features would look like with the current camera pose. Using optimization, they try to converge to a pose where the predicted features are similar to the observed features of the street view.

We note that none of the existing street-to-satellite image metric localization methods learns explicit local correspondences between the two views. Besides, Shi et al.'s method is the only method that uses the azimuth prior. However, they use the prior to map satellite features to a street view with a hardcoded transformation and not to learn explicit local correspondences between the two views.

2.2.2. Method Comparison

The proposed street-to-satellite image metric localization methods discussed in the previous subsection are compared in Tab. 2.1. All four methods can do location estimation. However, only the methods of Xia et al. [92] and Shi et al. [72] can do orientation estimation. The method of Shi et al. directly outputs a 3-DoF pose, while the method of Xia et al. does orientation estimation indirectly. Xia et al. compared the activations of the activation map before the final softmax operation for different orientations of the street image. They noted that the orientation can be predicted by taking the orientation of the street image associated with the highest activation. So multiple forward passes of the network are needed to estimate the orientation.

Both the graph matching-based approach of Verde et al. [82] and the method of Xia et al. output a probability distribution for the location of the street camera in the satellite image. The method of Verde et al. is the only method where objects are explicitly matched between the two views, and the graph can be used to gain insights into how the method's prediction was produced. However, Xia et al.'s method splits the satellite feature map into a grid and computes local satellite features. Therefore, that method is also considered to make explicit local matches between the two views. Finally, the method of Shi et al. requires the height of the street camera above the ground plane and the camera parameters.

Table 2.1: Comparison of the existing street-to-satellite image metric localization methods. *Dense output* means a multi-modal output such as a probability distribution.

Method	Year	Location estimation	Orientation estimation	Dense output	Explicit local matches	Camera parameters
Verde et al. [82]	2020	✓	X	✓	✓	X
Zhu et al. [107]	2021	✓	X	X	X	X
Xia et al. [92]	2022	✓	✓(indirect)	✓	✓(satellite)	X
Shi et al. [72]	2022	✓	✓	X	X	✓

2.2.3. Datasets

Zhu et al. [107] proposed the VIGOR dataset for street-to-satellite image metric localization. The street image locations can occur throughout the entire satellite image. For each street and satellite image pair, the location of the street image in the satellite image is known in pixels. In addition, the ground resolution is available, which can be used to convert a distance in pixels to meters. This allows for meter-level evaluation.

The supplemented version of the Oxford RobotCar dataset proposed by Xia et al. [90] can also be used for image retrieval and metric localization. This dataset has a satellite map of Oxford (United Kingdom) so that satellite images can be extracted. The street images of the Oxford RobotCar dataset are normal camera images and, therefore, have a limited horizontal field of view. The original street images have a horizontal field of view of 66° [47], but Xia et al. [92] cropped the street images to a field of view of 60°.

Shi et al. [72] extended the KITTI [25] and Ford multi-AV dataset [1] in a similar way as Xia et al. did for the Oxford RobotCar dataset. The original KITTI and the Ford multi-AV dataset both contain street images captured by a vehicle-mounted camera but without satellite maps. Therefore, Shi et al. supplemented them with the corresponding satellite images. Tab. 2.2 shows a comparison between the VIGOR, Oxford RobotCar, KITTI, and Ford multi-AV datasets.

Table 2.2: Comparison between the street-to-satellite image metric localization datasets. The data in the table is based on the sources cited for a dataset. The street and satellite image resolutions are not the raw resolutions but the resolutions used for training the methods. The ground resolution is based on the specified satellite image resolution in the table, and the unit is meters/pixel.

	VIGOR [107]	Oxford RobotCar [46, 47, 90]	KITTI [25, 72]	Ford multi-AV [1, 72]
Number of street images	238,696	23,854 [92]	30,970	50,441
Street panorama images	✓	X	X	X
Street image resolution	320 x 640	154 x 231 [92]	256 x 1024	256 x 1024
Satellite image resolution	640 x 640	800 x 800 [92]	512 x 512	512 x 512
Ground resolution	0.114	0.092 [92]	0.200	0.220
Generalisation	same-area cross-area	same-area across time	same-area cross-area	same-area

2.3. Local Image Feature Matching

Feature-based image matching methods use local image features, i.e., local image descriptors, to find correspondences between images. Usually, these methods have a pipeline consisting of three components: (1) feature detection and description, (2) feature matching, and (3) geometric verification [44]. Feature detection extracts distinctive points from an image, while feature describing encodes image information. Feature matching compares the information of a set of detected and described points to a reference set using a similarity measure. Geometric verification determines how well the matching satisfies one or more geometric constraints according to a metric. The result of the geometric verification can be a subset of the matches that can then be used for, for example, pose estimation [62] or determining a score for image retrieval [15, 50].

Many methods do the detecting and describing of the image interest point simultaneously to improve the final matching performance [18, 23, 51, 70, 94]. These are also called *detect and describe* methods. The core challenge of these methods is to make the whole process differentiable and trainable [44]. The difficulty of learning to detect and describe simultaneously is that at the beginning of the training, both the detections and descriptors are relatively poor, so the detector cannot use good descriptions and vice versa. Therefore, the network may not learn anything.

2.3.1. Detect and Describe Image Interest Points

Image interest points are 2D locations in an image that have properties like repeatability, invariance, robustness, and efficiency [18, 23, 44]. Repeatability implies that a relatively high ratio of the detected interest points can be matched with the detected points of the other view. Invariance and robustness mean that the detected interest points can be matched despite different lighting conditions and viewpoints. Lastly, efficiency means that the detection of interest points is computationally efficient so that it can be incorporated into real-time applications, for example.

In general, a distinction is made between two types of keypoints, namely (1) semantic keypoints and (2) interest points. A semantic keypoint is a point with semantic meaning for an object in the image [27, 35]. Image interest points are low-level points without a clear semantic meaning, such as a corner point of a cube or an endpoint of a line segment. This ensures that an image usually has a relatively large number of interest points, and the semantic meaning of those points is ill-defined, which makes annotating interest points in an image difficult. In this work, we only consider image interest points.

Detection: The detection of image interest points has a long history predating deep learning and several well-known methods have been proposed including SIFT [43], FAST [59], and ORB [60]. Inspired by handcrafted feature detectors, a general solution for CNN-based detection is constructing a response map to search for interest points. A response map is a map of typically the same spatial size as the image with interest point scores for each pixel location. Interest points can then be obtained by using the points with a score exceeding a certain threshold. Interest points close to each other can be filtered using non-maximum suppression. The creation of the response maps can be done in a supervised [83, 94, 101], self-supervised [23, 100], and unsupervised manner [5, 18, 51].

The task of detecting interest points is often formulated as a regression problem. Supervised methods have demonstrated the benefits of anchors that can guide the training. For example, these anchors can be obtained with SIFT [43]. However, the performance of such methods may be limited by the creation of the anchors because it is difficult to define an anchor, and the method will not learn to detect interest points in areas where there are no anchors [5]. Self-supervised and unsupervised methods train detectors without human annotations by using image transformations and geometric constraints [5, 18, 23, 51, 100].

Description: After image interest points have been detected, a descriptor must be assigned to each point in order to establish feature correspondence correctly and efficiently across the views [44]. In other words, the descriptors encode the local image information in a discriminative vector so that two matching features have a similar encoding and two non-matching features have a dissimilar encoding. The procedure for creating descriptors can be divided into three steps, namely (1) local low-level feature extraction, (2) spatial pooling, and (3) feature normalization [13, 43, 60]. First, the low-level information of the local image region of the interest point must be encoded. Second, the local patch is divided into several parts, and the information is pooled in each part and then concatenated by using pooling methods [43, 48, 60, 80]. Finally, a descriptor is obtained from the normalized results of the pooled local information.

Detect and describe methods: Yi et al. [94] proposed LIFT to simultaneously implement interest point detection, orientation estimation, and feature description by end-to-end CNN networks. LIFT can be regarded as a trainable version of the SIFT method.

DeTone et al. [23] noted that LIFT is trained with supervised learning and uses patches to detect interest points, meaning many patches must be created to detect interest points for an entire image. Therefore, they proposed SuperPoint, a fully convolutional model that can detect and describe interest points for full-sized images. SuperPoint is trained with a self-supervised learning framework that consists of three steps. Step 1 is to train a network called MagicPoint on a synthetic dataset with simple geometric shapes whose ground truth interest points are available. Step 2 is to use homographic adaptation where the input of the trained MagicPoint is varied, i.e., different homographies, to get pseudo ground truth interest points. Step 3 is the joint training of the detector and descriptor using the pseudo ground truth interest points from step 2. UnsuperPoint [18] eliminated steps 1 and 2 of SuperPoint's framework, which means that a synthetic dataset is no longer required. The concept is that interest points emerge naturally during training. Because steps 1 and 2 are not used, pseudo ground truth interest points are unavailable. This is solved by forming point pairs by mapping detected points from the source view to the target view and searching for the nearest neighbor in the target view for each point. This does require a known transformation between the two views.

Ono et al. [51] introduced a fully convolutional model called LF-Net that does detection and description sequentially rather than simultaneously with two or more branches, such as SuperPoint and UnsuperPoint. LF-Net also works with full-sized images and is trained with self-supervision. The detector generates a scale-space score map along with orientation estimates for detecting interest points. From the score map, the K highest scoring interest points and their image locations are extracted. These are used to extract a patch for each interest point from the normalized image. The patches are used to create descriptors. RF-Net is proposed as an improvement of LF-Net. It has a receptive field-based detector which generates more effective scale-space score maps than the detector of LF-Net. LF-Net uses ResNet [28] to produce feature maps from the input image and generates response maps for different scales by resizing the feature maps. This ensures that the response on each map has a large receptive field. On the other hand, RF-Net creates response maps using different receptive fields.

Barroso-Laguna et al. [5] combined handcrafted and learned CNN filters within a shallow multi-scale architecture called Key.Net. The handcrafted filters provide anchor structures for localizing, scoring, and ranking repeatable features that are fed to learned filters, and they reduce the number of learnable parameters of the network. The design of the handcrafted filters was inspired by the success of Harris [26] and Hessian [7] detectors, which used first and second-order derivatives to compute the salient corner responses. The detector is trained with weak supervision as only the ground truth transformation between two images of an image pair is used during training.

2.3.2. Match Image Interest Points

The matching task aims to establish the correct image pixel or point correspondences between two images using feature detection and description. This task has played a significant role in the entire image matching pipeline [44]. Descriptor matching followed by mismatch removal, also called indirect image matching, casts the matching task into a two-stage problem. First, preliminary correspondences are established by measuring the distance between descriptors with a similarity measure. Several common strategies, including fixed threshold, nearest neighbor, mutual nearest neighbor, and nearest neighbor distance ratio, are available for constructing these correspondences [44]. Second, the incorrect matches are removed using geometrical constraints. Usually, the classic RANSAC algorithm [24], or a modified version of it, is used to remove incorrect matches [44].

Learning from points: Point-based learning, particularly for feature matching, has only been introduced in recent years because using deep learning techniques for point data is more difficult than for raw images due to the unordered structure and dispersed nature of sparse points [44]. Besides, operating and extracting the spatial relationships among points, e.g., neighboring elements and relative positions, is challenging. However, using deep learning techniques to solve points-based tasks has received increasing attention [44, 63]. These techniques can be roughly divided into *parameter fitting* [9, 56] and *point classification or segmentation* [45, 54, 55, 95, 102]. The parameter fitting techniques are inspired by the RANSAC algorithm and aim to estimate the transformation model, such as epipolar geometry [9], using a data-driven optimization strategy with CNNs. On the other hand, point classification or segmentation techniques tend to train a classifier to identify the correct matches from the set of preliminary correspondences. In general, the parameter fitting and point classification techniques are trained jointly to enhance performance.

For parameter fitting, Brachmann et al. [9] proposed a differentiable RANSAC called DSAC for trainable fundamental matrix estimation. DSAC is based on reinforcement learning and has probabilistic selection instead of the deterministic hypothesis selection of RANSAC to decrease the expected loss and optimize the learnable parameters. Subsequently, Ranftl and Koltun [56] proposed a trainable method for fundamental matrix estimation from noise. Similar techniques as DSAC have been proposed by Brachmann and Rother [10] and Kluger et al. [34]. Brachmann and Rother proposed NG-RANSAC, a robust estimator using learned guidance of hypothesis sampling. It uses the inlier count as a training objective to facilitate self-supervised learning of NG-RANSAC. Kluger et al. presented CONSAC, a robust estimator for fitting multiple parametric models of the same form to noisy measurements.

For mismatch removal, several learning-based methods have been proposed in recent years. Moo Yi et al. [95] proposed LFGC, which aims to label the preliminary correspondences as inliers and outliers, and Ma et al. [45] proposed a general framework to learn a two-class classifier for mismatch removal called LMR. Sarlin et al. [63] proposed SuperGlue, a neural network that matches two sets of local image features by jointly finding correspondences and rejecting non-matchable points. Assignments are estimated by solving a differentiable optimal transport problem, whose costs are predicted by a graph neural network [66]. Compared to traditional, handcrafted heuristics, their technique learns priors over geometric transformations and regularities of the 3D world through end-to-end training from image pairs.

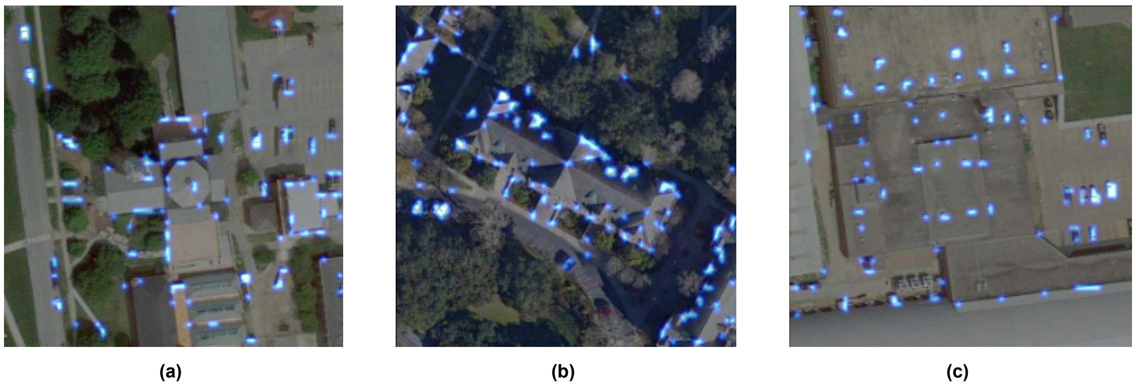


Figure 2.2: Visualization of the detected interest points with RK-Net [39] for three images of the University-1652 dataset [104]. The figures were taken from Lin et al. [39].

2.3.3. Local Image Features for Street-to-Satellite Image Matching

Cross-view matching methods use global image descriptors to match the street and satellite images (see Sections 2.1 and 2.2). However, Lin et al. [39] proposed RK-Net to jointly learn a discriminative representation and detect salient interest points with a single network. The authors introduced a unit subtraction attention module that automatically discovers representative interest points from feature maps and draws attention to the salient regions. The detected interest points are used for reweighing the feature map and not for direct matching with interest points of the other view. The unit subtraction attention module contains few learning parameters but yields significant performance improvement and can be plugged into different networks [39]. Fig. 2.2 shows visualizations of interest points detected by RK-Net.

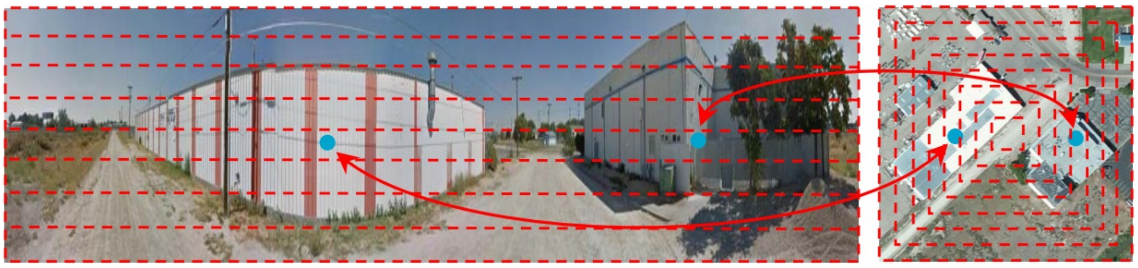
We note that detect and describe methods use one-to-one correspondences between images to train the detector and descriptor. SuperPoint uses homography to create pseudo ground truth interest points. In contrast, the viewpoint difference for street-to-satellite image matching is significantly larger than for multiple-view image matching. Therefore, homography cannot be applied to create pseudo image pairs. Besides, unsupervised learning cannot be used directly for detecting image interest points since it requires interest point pairs during training. For cross-view matching, it is difficult to get these as only the azimuth prior is available. So for a point in the street image, the azimuth angle can be used to determine in the satellite image the azimuth direction on which a possible matching point lies and vice versa. However, it is unknown if (1) there is a matching point and (2) which point on the line is the possible matching point. Besides, due to the large viewpoint difference between the two views, different sides may be visible in the images of objects. So for those objects, direct matches do not exist. Lastly, the azimuth prior can be used as a constraint for filtering preliminary correspondences, but learning-based methods for interest point matching also depend on ground truth point pairs. Therefore, the methods described in this section cannot directly be used for street-to-satellite image metric localization.

2.4. Azimuth Prior

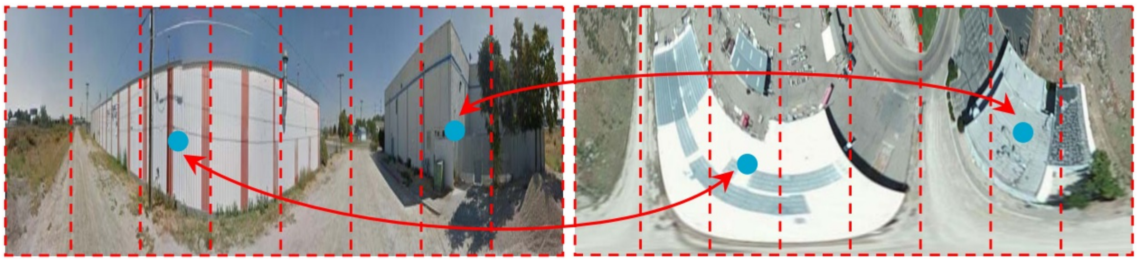
The geometric relationship between the street and the satellite view, i.e., the *azimuth prior*, has been noted by many cross-view matching works [37, 42, 73, 74, 75, 86, 106]. The relationship implies that every vertical line in the street image has a corresponding azimuth direction in the satellite image.

Several cross-view matching approaches have been proposed that map the street image using stereo vision and the camera parameters to a bird's-eye view so that the street image looks more like the satellite image [49, 68, 84]. However, these methods still rely on common object features between the two views. For example, road signs on the road can be used for matching, but objects such as buildings can still not be used. In addition, stereo vision is usually unavailable for cross-view matching [29, 73, 79, 107].

Image retrieval methods have been proposed that use the azimuth prior to transform an image from one view to another view (see Subsection 2.1.4). Shi et al. [73] assumed that the street camera location is at the center of the satellite image and proposed the polar transformation for transforming a satellite image into a synthetic street image. Later, Shi et al. [75] proposed the projective transform, an improved version of the polar transformation in which the camera parameters are used to create a more realistic synthetic street image. In contrast, Li et al. [37] proposed the inverse polar transformation for



(a) Sequential slicing



(b) Column slicing

Figure 2.3: The two slicing methods proposed by Wang et al. [86]. The figure was taken from Wang et al. [86].

mapping a street image to a synthetic satellite image. All these image transformations are hardcoded and exhibit image distortions because the depth is unknown, and there is a lack of object information for mapping to the other view.

Liu and Li [42] used the azimuth prior for providing orientation information as additional input to the network. Yang et al. [93] noted the azimuth prior but did not use hardcoded orientation embedding. Instead, they use learnable embeddings to let the network learn the geometric relationship via the training objective.

Wang et al. [86] proposed partitioning the feature maps that follow from the feature extractors into slices. The slicing ensures that only information from the feature maps likely to show the same portion of the local surroundings is matched. They proposed two slice methods for street-to-satellite image matching, namely (1) sequential partitioning and (2) column partitioning. Fig. 2.3a shows the sequential partitioning. This partitioning is based on the observation that objects higher in the street image are generally further away from the street camera. It is assumed that the street camera was located at the center of the satellite image. A polar-transformed satellite image is used for column partitioning (see Fig. 2.3b). This partitioning is based on the azimuth prior, where the azimuth range is divided into a discrete number of blocks.

The azimuth prior has also been used in fields related to cross-view matching. Shi et al. [71] used the prior for synthesizing street panoramas using satellite imagery. Saha et al. [61] used the prior for mapping street images to segmented overhead maps, i.e., the bird's eye view of the world. They assume a one-to-one correspondence between a vertical scanline in the street image and rays passing through the camera location in the overhead map. This lets them formulate map generation from an image as a set of sequence-to-sequence translations.

We note that the azimuth prior has been used to transform an image from one view to another. In addition, the prior has been used to adjust network inputs so that the model can use hardcoded orientation information directly or learn the geometric relationship between the two views. Also, the prior has been used to split feature maps so that only information expected to show the same part of the local surroundings is matched. Finally, the prior has been used for synthesizing realistic street panoramas using satellite imagery and mapping street images to segmented overhead maps. However, none of the methods used the prior as a weakly supervised signal for learning correspondences between the two views.

2.5. Our contributions

Existing street-to-satellite image metric localization methods do not learn explicit local correspondences between the street and satellite view and do not use the azimuth prior for improving the localization performance (see Section 2.2). Besides, we note that local image feature matching methods use one-to-one correspondences during training, while these are unavailable for street-to-satellite image matching (see Section 2.3). Lastly, the azimuth prior has been used to improve the performance of street-to-satellite image retrieval methods and related fields for mapping images to another view (see Section 2.4). However, the prior has not been used for learning local correspondences between the two views.

We use the azimuth prior as a weakly supervised signal to learn local correspondences between the street and satellite view for street-to-satellite image metric localization. The contributions of this work can be summarized as follows:

1. This work proposes SliceNet, the first end-to-end local image feature matching-based network for street-to-satellite image metric localization, which can predict a dense pose distribution for the street camera in the satellite image. SliceNet sets a new state-of-the-art for localization on the VIGOR dataset when generalizing in the same area and across areas.
2. This work quantitatively and qualitatively investigated the role of horizontal and vertical street image slices in localizing the street camera in the satellite image. It has been shown that the localization performance can be increased by using more horizontal slices. Moreover, it has been demonstrated that SliceNet achieves a significantly higher inference speed than the baselines.
3. This work discovered that the original ground truth labels of the VIGOR dataset contained an error due to the use of an incorrect ground resolution. New ground resolutions have been determined, and the ground truth labels have been corrected. Moreover, all existing baselines for the VIGOR dataset have been trained and evaluated with the corrected ground truth labels.

3 Methodology

This chapter describes the methodology used to answer the research questions posed in the Introduction (see Chapter 1). Section 3.1 motivates the need for learning sparse local image feature matching for street-to-satellite image metric localization and Section 3.2 presents the baselines against which we compare. Subsequently, Section 3.3 describes the plan for achieving sparse local image feature matching. After that, the proposed method is described in Section 3.4. This chapter ends with Section 3.5 that compares the proposed method with the baselines.

3.1. Problem Definition

The objective of street-to-satellite image metric localization is to determine the street camera location $X_c = (h_c, w_c) \in \mathbb{R}^2$ in the satellite image $I_s \in \mathbb{R}^{H_s \text{x} W_s \text{x} C}$ using the street image $I_g \in \mathbb{R}^{H_g \text{x} W_g \text{x} C}$. This image location can be converted to the longitude and latitude of the street camera using the geo-tag and ground resolution, i.e., the number of meters per pixel, of the satellite image. The variables H, W, and C indicate an image's height, width, and the number of channels, respectively. The *ground-level* street view and the satellite view are indicated by *g* and *s*, respectively, to avoid confusion.

For each street and satellite image pair, a ground truth street camera pose $\xi_{GT} = (h_{GT}, w_{GT}, \theta_{GT})$ is known during training at which the street image has been captured with $h_{GT} \in [0, H_s]$, $w_{GT} \in [0, W_s]$, and $\theta_{GT} \in [0°, 360°)$. Both h_{GT} and w_{GT} have the unit pixels. The street camera orientation indicates the angular difference of the vertical centerline of the street image to North in the satellite image. Here counterclockwise is defined as positive in the satellite image. It is assumed that the horizontal field of view $\phi \in (0°, 360°]$ of the street image is known. Fig. 3.1 shows the defined variables for an image pair.

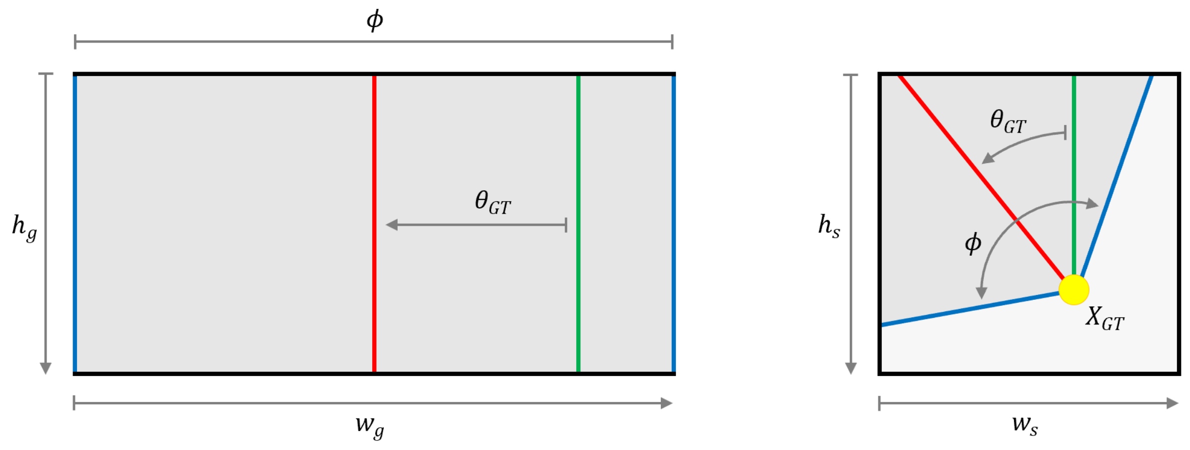
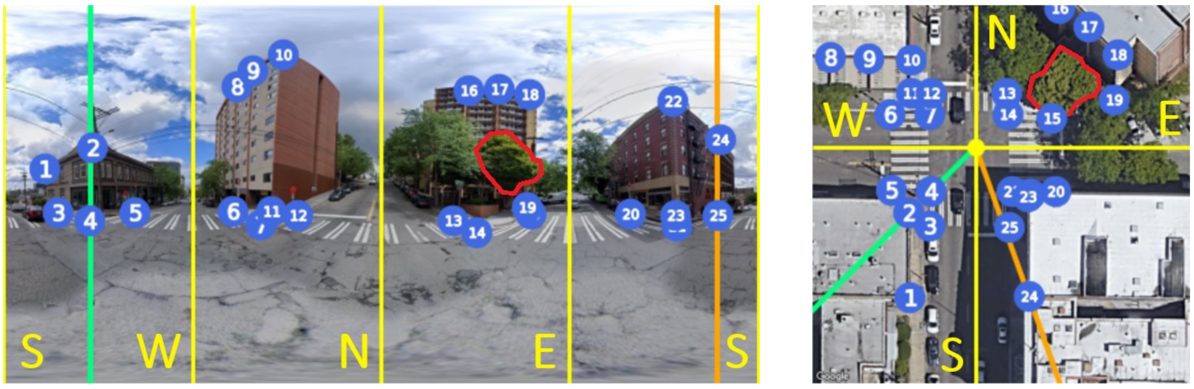


Figure 3.1: A schematic street and satellite image pair with the ground truth street camera pose indicated. The left and right images are the street image and the satellite image, respectively. The yellow dot in the satellite image indicates the ground truth location X_{GT} at which the street image has been captured. The satellite image's red line indicates the street image's viewing direction. This line is the vertical centerline of the street image. The ground truth orientation θ_{GT} is defined as the angular difference between the vertical centerline of the street image (red line) and North in the satellite image (green line). For this orientation, counterclockwise is positive (satellite view). The blue lines indicate the edges of the horizontal field of view ϕ of the street image. The variables h and w indicate the height and width of an image location, respectively.



(a) Street (360° panorama) image **(b)** Satellite image

Figure 3.2: A street and satellite image pair with manual annotations. Each annotated point in the street image has a corresponding annotation, i.e., the same number, in the satellite image. The red contour indicates an entire tree. The image pair is part of the VIGOR dataset [107].

3.1.1. One-to-One Matches between Views

Street camera localization requires matching the content of the street image with that of the satellite image. The street and the satellite image both show the local surroundings of the street camera, and thus *one-to-one matches* can be made between the two images. Fig. 3.2 shows an image pair with manual annotations for the corners of the buildings and the sidewalks. Each annotated point in the street image has a matching annotation in the satellite image. Apart from these points, one-to-one matches could also be made for other objects, such as the crosswalk.

For trees, it is more challenging to make one-to-one matches because of the difference in viewpoint, e.g., the street image shows the side view of the trees, and the satellite image shows the bird's-eye view. In addition, the images were not created simultaneously, so each tree in the street image looks slightly different than in the satellite image. However, an entire tree can be matched, as indicated by the red contour in both images.

There are also parts of the local surroundings that can only be seen in one of the two images. For example, the street image shows the sides of the buildings, while the satellite image shows the roofs of the buildings. Besides, objects visible in the street image in the distance may not be seen on the satellite image due to the limited area of the local surroundings shown by the satellite image. Objects only visible in one of the two images cannot be matched. Therefore, only sparse one-to-one matches are possible between the two views, i.e., only a subset of the visible objects can be matched.

3.1.2. Determine Street Camera Location and Orientation

Street-to-satellite matches can be used to determine the street camera location in the satellite image. One way the matches can be used to determine the location is by combining matches, i.e., one-to-one pairs. The geometric relationship [42, 73, 74, 106] between the street and the satellite view implies that every vertical line in the street image has a corresponding azimuth direction in the satellite image (see Fig. 1.3). Thus, two pairs whose street points approximate a vertical line in the street image construct a line in the satellite image that passes through the location of the street camera.

Fig. 3.2 shows two one-to-one pairs, i.e., the green and orange lines. The points labeled 2 and 4 are approximately on the green vertical line in the street image, and this line passes through the location of the street camera in the satellite image. The same applies to the orange line formed by the points with labels 24 and 25. Intersections of constructed lines can be used as a street camera location estimation. Subsequently, the orientation of the street image can be estimated by determining the angular difference between the azimuth direction of the street image's vertical centerline and North in the satellite image.

For objects such as trees for which the two views show different parts of the object, it is unclear which location should be used for the matching. Therefore, matching those objects requires a local descriptor representative of an entire object, i.e., of multiple pixels. An example would be all pixels within a red contour in Fig. 3.2. A location within the contour can still be used to construct a line with another one-to-one pair. However, the uncertainty about the line parameters is then more significant.

3.1.3. Learn Local Correspondences

Annotated one-to-one correspondences between the two views are unavailable for street and satellite image pairs. Thus, we do not have labels that can be used to learn the one-to-one matches. Therefore, an alternative must be found so a network can still learn the local correspondences. This is described in Section 3.3, but first, Section 3.2 describes what baselines are used and how these methods determine the street camera location.

3.2. Baselines

Section 3.1 described the street-to-satellite image metric localization task and explained how sparse local image features could be used to determine the street camera location. Existing methods [72, 92, 107] do not use local image features for predicting the street camera location, and a total of three end-to-end methods have been proposed: (1) the cross-view regression method of Zhu et al. [107], (2) the multi-class classification method of Xia et al. [92], and (3) the optimization-based method of Shi et al. [72]. The cross-view regression method and the multi-class classification method are also known as the *CVR* model and the *MCC* model, respectively [92].

3.2.1. Required Update Frequency

In the introduction (see Chapter 1), bounds for the localization error for autonomous vehicles have been stated. Besides the localization accuracy, the update frequency of a method is also important because a lag in the position update leads directly to further uncertainty in the localization [58]. This uncertainty is predominantly in the longitudinal direction.

An update frequency equal to 100 hertz or greater appears to be the appropriate update rate for freeway roads and local streets [58]. This requirement does not mean that the image metric localization method must achieve this update frequency because the vision-based predictions could be combined with temporal filtering, for example. However, we do consider the inference speed of the existing methods for choosing our baselines.

Shi et al. noted that the pose optimization runtime for a single image pair is around 500 milliseconds for their method. This long runtime is because the optimization uses iterations to determine the camera pose. In contrast, Xia et al. measured that a forward pass of the CVR model and the MCC model takes around 20 and 30 milliseconds, respectively. In addition, the optimization-based method of Shi et al. requires the height of the street camera above the road to be known, while the CVR model and the MCC model do not need this height. Consequently, it has been decided to use the CVR model and the MCC model as our baselines because these methods are significantly faster than Shi et al.'s method, and they do not require additional information about the street camera for predicting the location.

3.2.2. Regression Baseline

The CVR model can localize the street camera in a satellite image in a coarse-to-fine manner by first doing image retrieval and then predicting the location of the street image in the retrieved satellite image (see Subsection 2.2.1). The architecture of the CVR model consists of a street and satellite branch with each a VGG16 [76] as feature extractor and a SAFA module [73] as descriptor network (see Fig. 3.3).

For the street branch, feature extractor 1 computes the street feature map F_g using the street view

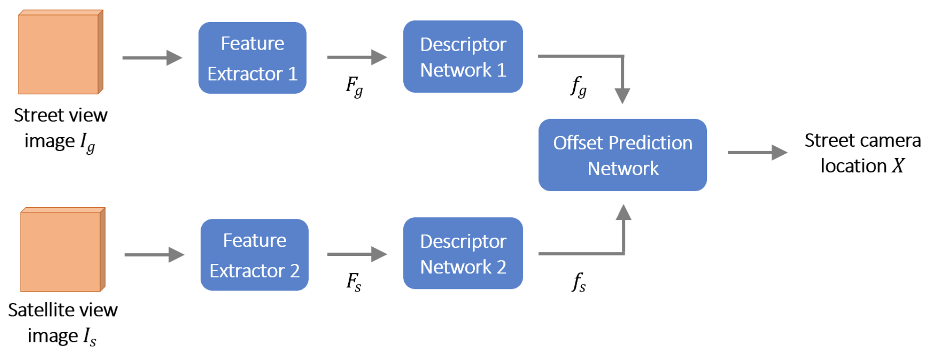


Figure 3.3: The architecture of the CVR model [107]. The architecture consists of a street and satellite branch with each a VGG16 [76] as feature extractor and a SAFA module [73] as descriptor network. The output of the two branches is concatenated and fed into a multilayer perceptron that predicts the location of the street camera in the satellite image.

image I_g as input, and descriptor network 1 uses this feature map to make the street global image descriptor f_g. Similarly, the satellite branch uses the satellite view image I_s to compute the satellite feature map F_s and the satellite global image descriptor f_s. The two global descriptors are used to do image retrieval. Subsequently, metric localization is performed by predicting an offset for the street camera location with respect to the satellite image center. The offset prediction is made with a multilayer perceptron, and the input consists of the concatenated global descriptors.

We only employ the CVR model for the metric localization task but keep its proposed architecture. However, we will not train and use it for image retrieval because we assume that the matching satellite image is already available (see Section 3.1). For training the model, the standard mean squared error loss, i.e., the L2 loss, is used to minimize the error between the predicted offset and the ground truth offset. More details about the CVR model can be found in the work of Zhu et al. [107].

3.2.3. Classification Baseline

The MCC model computes a dense distribution for the street camera location in the satellite image (see Subsection 2.2.1). The architecture of the MCC model consists of a street and satellite branch combined with a decoder (see Fig. 3.4).

The street branch is identical to the street branch of the CVR model (see Fig. 3.3). However, the satellite branch splits the satellite feature map F_s in a grid and computes for each grid cell a descriptor. The street global descriptor is fused with the satellite descriptors by calculating the cosine similarity between the street global descriptor and the satellite grid descriptors. The decoder uses the resulting matching map in combination with the satellite descriptors to predict a probability distribution for the street camera location. Residual connections are used so the decoder can use intermediate feature representations from the satellite branch. More details about the MCC model can be found in the work of Xia et al. [92].

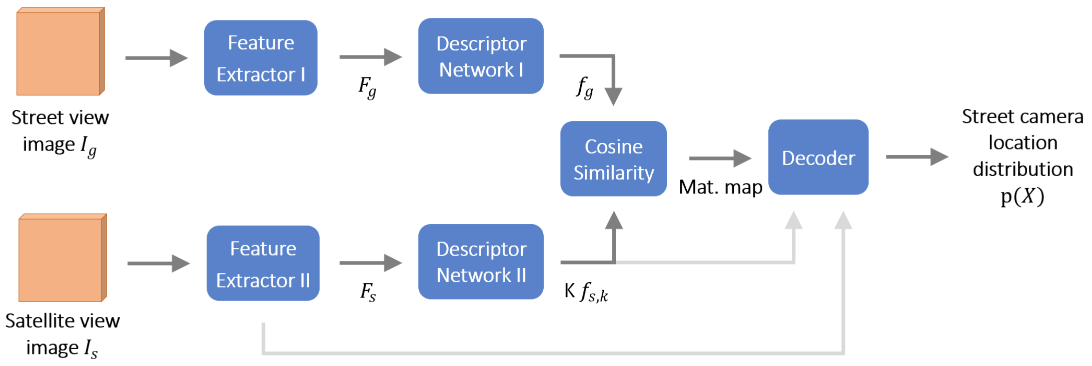


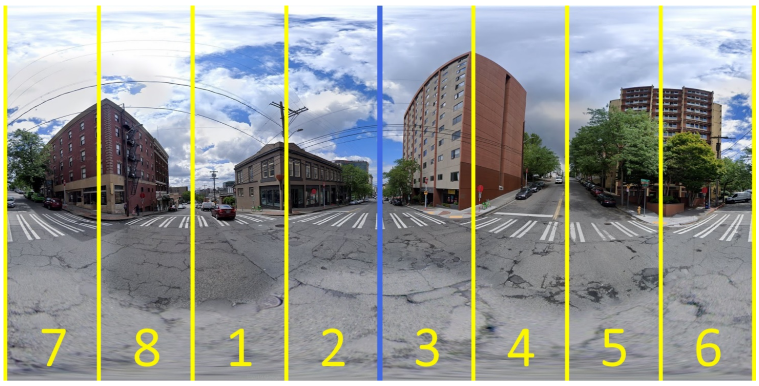
Figure 3.4: The architecture of the MCC model [92]. The architecture consists of a street and satellite branch with each a VGG16 [76] as feature extractor and a SAFA module [73] as descriptor network. The street branch computes the street global descriptor f_g, while the satellite branch splits the satellite feature map into a grid and computes K satellite descriptors $f_{s,k}$, i.e., for each grid cell one. The output of the two branches is matched with the cosine similarity. The decoder upsamples the matching map, denoted by *Mat. map*. Light gray arrows indicate residual connections between the satellite branch and the decoder.

3.2.4. Baseline Comparison

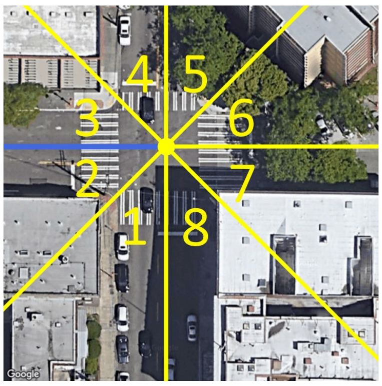
There are three main differences between the CVR model and the MCC model. First, the CVR model computes a single descriptor for the satellite image, while the MCC model splits the satellite feature map into a grid and computes a descriptor for each grid cell. Second, the CVR model does not match the two global descriptors explicitly but fuses the information from the two branches by concatenating the descriptors. In comparison, the MCC model measures the cosine similarity between the street global descriptor and the satellite descriptors. Third, the CVR model computes the street camera location offset with respect to the center of the satellite image. In contrast, the MCC model uses a decoder to predict a probability distribution for the street camera location.

3.3. Achieving Sparse Local Image Feature Matching

Section 3.1 described that we do not have labeled correspondences between the street and satellite view that can be used to learn sparse local image features. However, we do have the geometric relationship between the street and satellite image, i.e., the azimuth prior. This prior implies that every vertical line in the street image has a corresponding azimuth direction in the satellite image (see Fig. 1.3).



(a) Street (360° panorama) image



(b) Satellite image

Figure 3.5: A street and satellite image pair with eight image slices that have been made with horizontal slicing. Each street slice has a corresponding satellite slice, and the number of the slice indicates this. The street image is not aligned with the satellite image, i.e., the viewing direction of the street image (blue line) does not correspond with North in the satellite image. The image pair is part of the VIGOR dataset [107].

For the azimuth prior to be valid, we make three assumptions. First, it is assumed that the autonomous vehicle has no roll rotation and that the street camera is horizontally aligned. In other words, a vertical line in the street image corresponds to a vertical plane in the real world. Second, it is assumed that the image plane of the satellite camera is parallel to the road. Third, it is assumed that an orthographic correction has been applied to the satellite image. This means that all projection lines are parallel to each other and perpendicular to the projection plane. Thus, it ensures that the visible area on the satellite image is viewing distance invariant. The three assumptions usually apply to vehicle localization.

Based on the second and the third assumption, a vertical plane in the real world is projected as a straight line in the satellite image. Therefore, a vertical line in the street image (the first assumption) corresponds to a line in the satellite image, i.e., the azimuth direction.

3.3.1. Horizontal Slicing

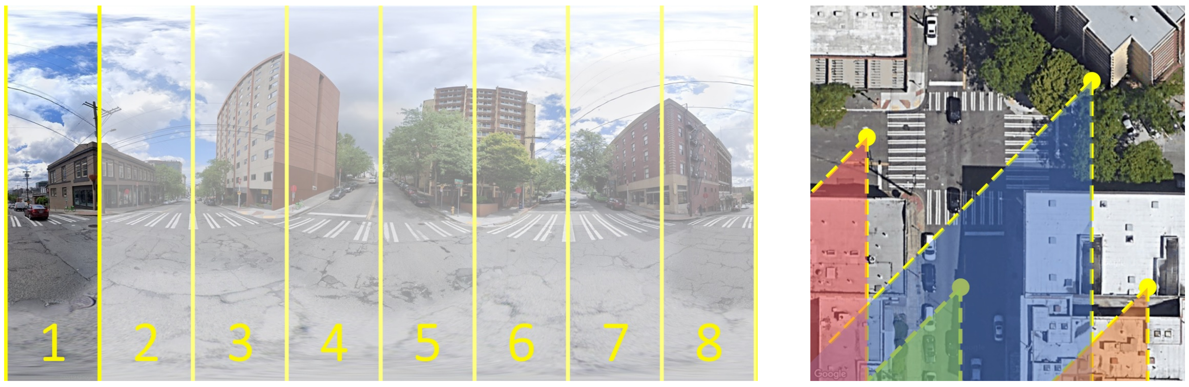
Image slices in the street and the satellite image can be constructed by exploiting the azimuth prior. Based on this prior, the horizontal field of view of the street image can be divided into N equal-sized image slices, i.e., rectangular image patches. Similarly, the satellite image can be divided into N pie part-like slices using the azimuth directions corresponding to the vertical lines used to create the street slices. Fig. 3.5 shows an image pair with eight slices per image.

This way of slicing is defined as *horizontal slicing* because the azimuth directions in the satellite image are parallel to the road, i.e., the *horizontal* ground plane. Each street slice has a corresponding satellite slice. Corresponding slices, i.e., matching slices, overlap in terms of the local surroundings they show, while non-matching slices have no overlap. However, non-matching slices can have semantic overlap if, for example, two slices both show a tree.

3.3.2. Orientation Estimation

The orientation of the street image with respect to the satellite image can roughly be determined when we match the street slices with the satellite slices. Fig. 3.5 is used to explain this. The viewing direction of the street image is the vertical line that separates slices 2 and 3 (blue line). By finding the matching satellite slices for these two street slices, the orientation of the street image can be determined. The slices are numbered, and thus it can be seen that the viewing direction of the street image is West in the satellite image, i.e., the street image orientation is 90° according to the definition of Fig. 3.1. For this image pair, slices are aligned so that each street slice only overlaps with one satellite slice. If this is not the case, it is impossible to determine the street orientation based on one-to-one slice matches exactly, and there is an error that depends on the size of the field of view associated with a single slice.

We hypothesize that we can learn orientation discriminative descriptors for the street and satellite slices where descriptors of matching slices are similar, i.e., close together in the feature space of the descriptors, and descriptors of non-matching slices are dissimilar. Creating the satellite slices requires the street camera pose, so this can only be done during the training of the network. If we can learn discriminative slice descriptors, then we can obtain a rough orientation estimate for the street image, as explained above.



(a) Street (360° panorama) image **(b)** Satellite image

Figure 3.6: A street and satellite image pair with eight horizontal street image slices, and four locations in the satellite image are used to create satellite image slices. Based on the orientation information of street slice 1, street slice descriptor 1 should only be matched with the descriptors of the colored slices in the satellite image. The matching for other street slice descriptors can be done similarly. The image pair is part of the VIGOR dataset [107].

For each image, N slice descriptors are created. Each street slice descriptor can be matched with all slice descriptors of the satellite image. However, we can use the spatial structure of the slicing, i.e., which slices are adjacent to a particular slice. Only permutations of the satellite slice descriptors that are consistent with the spatial structure of the slicing need to be used for matching, and there are N valid permutations, i.e., we can rotate the pie N times. The orientation estimate is not directly of value as we use the ground truth pose for the slicing. However, learning orientation discriminative descriptors is a step towards achieving sparse local image feature matching.

3.3.3. Location Estimation

The location of the street camera in the satellite image can be estimated if we use multiple locations in the satellite image for the slicing and choose the location for which the satellite slices match the best with the street slices, i.e., the highest similarity.

The street camera pose is required to make the correct satellite slices. Therefore, we cannot create these slices during inference. However, if we do the slicing for many locations in the satellite image, we can assume that one of the used locations is close to the camera location. This implies that we create N satellite slice descriptors for each location used in the satellite image. The location of the street camera can then be estimated by measuring the similarity of the street slice descriptors with the satellite slice descriptors of each location used and choosing the location with the highest similarity.

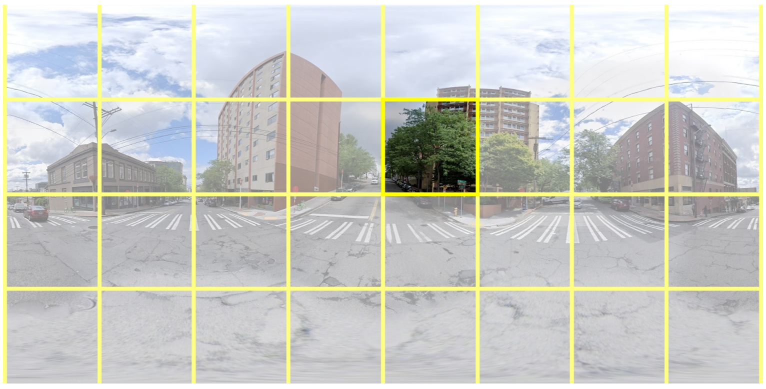
During training, the street camera pose can be used to determine the ground truth satellite slice descriptors and learn that those descriptors should have a higher similarity with the street slice descriptors than the descriptors of other locations. Whereas during inference, the slicing can be done for predetermined locations, and the true location can be estimated by measuring the similarity of the descriptors for each location.

We hypothesize that we can learn location discriminative satellite slice descriptors for which the satellite slice descriptors belonging to the street camera location are more similar to the street slice descriptors than the satellite descriptors of other locations in the satellite image. Fig. 3.6 shows an image pair with eight street slices, and four locations in the satellite image are used for the slicing. If the street image orientation is assumed to be known, i.e., the orientation is not estimated, then street slice descriptor 1 only needs to be matched with the descriptors of the four colored slices in the satellite image. The matching for other street slice descriptors can be done similarly.

3.3.4. Pose Estimation

For localization, it is assumed that the true orientation is known during inference. We can also drop this assumption by matching a street slice descriptor with all satellite slice descriptors instead of only with descriptors belonging to slices with a particular orientation. This can be seen as combining orientation estimation (see Subsection 3.3.2) and location estimation (see Subsection 3.3.3).

We hypothesize that we can learn pose discriminative satellite slice descriptors for which the satellite slice descriptors belonging to the street camera pose are more similar to the street slice descriptors than the satellite descriptors of other poses in the satellite image. During training, analogous to what


(a) Street (360° panorama) image

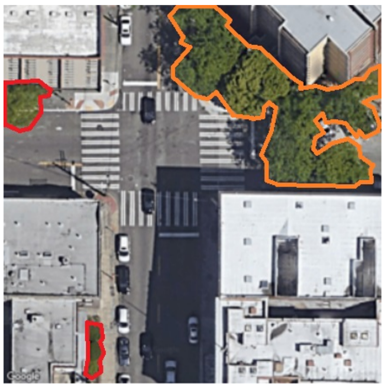

(b) Satellite image

Figure 3.7: A street and satellite image pair with eight horizontal image slices and four vertical image cells per slice in the street image. The designated cell in the street image shows a tree. In the satellite image, regions with trees are indicated by colored contours. The orange contour indicates the tree region to which the tree in the indicated street cell belongs, while the red contours are different tree regions. The image pair is part of the VIGOR dataset [107].

can be done for location estimation, we can use the street camera pose to determine the ground truth satellite slice descriptors and learn that those descriptors should have a higher similarity with the street slice descriptors than the descriptors of other poses. Whereas during inference, the slicing can be done for predetermined poses, and the true pose can be estimated by measuring the similarity of the descriptors for each pose. This way, localization can be done, and a rough estimate can be made for the orientation of the street image.

3.3.5. Vertical Slicing

We observe that a street slice can be divided into M vertical cells based on the semantic content visible in the slice. This way of slicing is defined as *vertical slicing* because the *vertical* direction of the street image is divided into *vertical* cells. In general, an image column consists of roughly four semantic parts, namely from bottom to top (1) the road, (2) the sidewalk or crosswalk, (3) buildings or trees, and (4) the sky. This observation can be used to structure the encoding of the street slice descriptor.

Fig. 3.7 shows an image pair where each horizontal slice of the street image is divided into four vertical cells. A separate descriptor can be created for each cell of a street slice, and the cell descriptors can be combined to get the slice descriptor. In contrast, the satellite slices cannot be divided in a similar way because objects can appear at any position in the satellite image. However, parts of the satellite image can be selected based on the type of content. By choosing particular objects in the satellite image for each vertical cell row in the street image, different descriptors can be created for a satellite slice. These descriptors can then be combined to create the satellite slice descriptor.

We hypothesize that we can learn to select objects in the satellite image that correspond to a specific vertical cell row and make discriminative cell descriptors that can be used for localizing the street camera in the satellite image. Splitting the street slice can be seen as an extension of the location estimation (see Subsection 3.3.3) and pose estimation (see Subsection 3.3.4).

3.3.6. Search for Specific Vertical Cell

The vertical slicing treats all cells of a particular vertical cell row equally. However, we can go one step further and search for matching content in the satellite image for a specific cell instead of an entire vertical cell row. This way, cells from the same vertical row can select different objects. Fig. 3.7 shows an image pair where each horizontal slice of the street image is divided into four vertical cells. The designated street cell shows a tree. In the satellite image, the orange contour indicates the matching tree region, while the red contours indicate non-matching tree regions. So for the designated cell, these three regions should be selected, while all other objects in the satellite image should not be used to create the corresponding satellite cell descriptor.

We hypothesize that we can learn to select objects in the satellite image that correspond to a specific vertical cell of a street slice and make discriminative cell descriptors that can be used for localizing the street image in the satellite image.

3.4. The SliceNet Architecture

The architecture of SliceNet is based on the pipeline of local feature matching-based methods (see Subsection 2.3). This pipeline consists of three components: (1) feature detection and description, (2) feature matching, and (3) geometric verification.

SliceNet extracts local features, i.e., slice descriptors, from the input street and satellite image. This is *feature detection and description*. Subsequently, the spatial structure of the horizontal slicing is used to combine the street and satellite slice descriptors into a street and satellite descriptor, respectively. The satellite descriptor depends on the used pose in the satellite image for the horizontal slicing (see Subsection 3.3.4). After the image descriptors have been formed, the descriptors are matched to determine the similarity for a pose in the satellite image. This is the *feature matching*. The street camera pose is estimated by taking the pose with the highest similarity. *Geometric verification* is not applied after the matching step since the geometric relationship between the two views is already used to create the image descriptors, so only descriptors that satisfy the geometric constraint are matched (see Subsection 3.4.4).

Fig. 3.8 shows the architecture of SliceNet. SliceNet has a Siamese-like architecture consisting of a street view and a satellite view branch. Both branches consist of three components: (1) a feature extractor that extracts deep features from the input image, (2) a cell attention module that selects vertical cell-specific features, and (3) a slice descriptor generator that creates slice descriptors given a feature map. The normalized slice descriptors that follow from a slice descriptor generator are combined to obtain the image descriptor of a view. This descriptor incorporates pose information. The symbols associated with the arrows in the figure are explained in the subsequent subsections.

We differentiate between two versions of SliceNet: (1) SliceNet Basic and (2) SliceNet Select. SliceNet Basic does not use cell attention modules, while SliceNet Select does use these modules. A cell attention module can select specific features from a feature map (see Subsection 3.4.2). Below, we first highlight the three components of the SliceNet architecture, and then the loss function and street camera pose prediction are discussed.

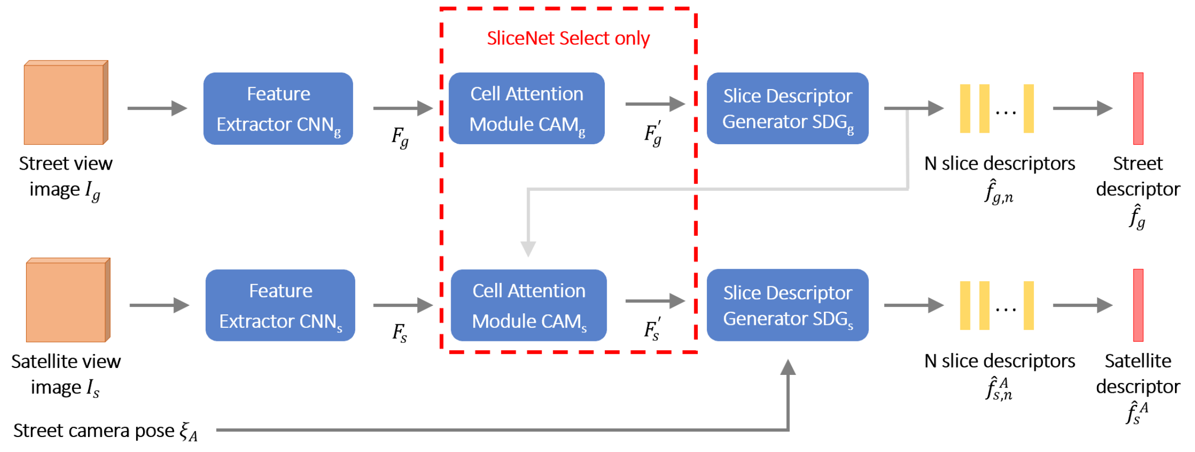

SliceNet Select only
Street view image I_g
Feature Extractor CNN$_g$
F_g
Cell Attention Module CAM$_g$
F_g'
Slice Descriptor Generator SDG$_g$
N slice descriptors $\hat{f}_{g,n}$
Street descriptor $\hat{f}_g$
Satellite view image I_s
Feature Extractor CNN$_s$
F_s
Cell Attention Module CAM$_s$
F_s'
Slice Descriptor Generator SDG$_s$
N slice descriptors $\hat{f}^A_{s,n}$
Satellite descriptor $\hat{f}^A_s$
Street camera pose ξ_A

Figure 3.8: The Siamese-like architecture of SliceNet. The architecture consists of two branches: a street view and a satellite view branch. Both branches contain a feature extractor, a cell attention module, and a slice descriptor generator. The feature extractor creates a feature map given an input image, the cell attention module reweighs the feature map, and this reweighted feature map is used in the slice descriptor generator to create slice descriptors. The normalized slice descriptors are combined to form an image descriptor. SliceNet Basic does not use cell attention modules, while SliceNet Select does use these modules. The light gray arrow indicates the cross branch connection.

3.4.1. Component 1: Feature Extractor

In both branches, a CNN is used as feature extractor. The feature extractors of the street and the satellite view branch are denoted by CNN$_g$ and CNN$_s$, respectively, and are used to obtain the street feature map $F_g =$ CNN$_g(I_g)$ and the satellite feature map $F_s =$ CNN$_s(I_s)$. The street and the satellite feature maps have a shape of $H_g^{'}$ x $W_g^{'}$ x $C^{'}$ and $H_s^{'}$ x $W_s^{'}$ x $C^{'}$, respectively. Both feature maps have the same number of feature channels. However, the spatial size can differ depending on the input images' spatial size. The stride of the feature extractors causes the spatial size of a feature map to be equal to or smaller than that of an input image. For a fair comparison with the baselines (see Section 3.2), we employ the same CNN as the baselines use, i.e., a VGG16 [76].

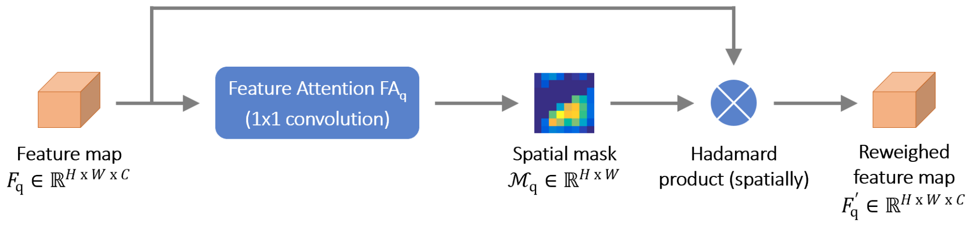

Feature map
$F_q \in \mathbb{R}^{H \times W \times C}$
Feature Attention FA$_q$
(1x1 convolution)
Spatial mask
$\mathcal{M}_q \in \mathbb{R}^{H \times W}$
Hadamard product (spatially)
Reweighed feature map
$F_q^{'} \in \mathbb{R}^{H \times W \times C}$

Figure 3.9: Feature Attention Mechanism. A spatial mask is obtained by applying 1x1 convolution to the input feature map followed by a sigmoid activation function. The feature map is multiplied with the mask to obtain the reweighed feature map.

3.4.2. Component 2: Cell Attention Module

The cell attention modules of the street and the satellite view branch are denoted by CAM$_g$ and CAM$_s$, respectively. These modules enable SliceNet to select important features in a feature map (see Fig. 3.9). Unimportant features are multiplied by relatively small values, i.e., close to zero, while important features are multiplied by a relatively large value, i.e., close to one. As a result, important features will have more impact on a slice descriptor because the features of a slice are averaged to compute the slice descriptor (see Subsection 3.4.3). The cell attention module has been inspired by the work of Cai et al. [14]. They proposed an attention module that uses feature attention to reweigh both the street and the satellite feature map to emphasize visually salient features for street-to-satellite image retrieval.

Street Branch

The street cell attention module CAM$_g$ uses 1x1 convolution to obtain the spatial mask $\mathcal{M}_g^{att}$ with shape $H_g^{'}$ x $W_g^{'}$. Subsequently, the Hadamard product is used to obtain the reweighed street feature map,

$$F_g^{'} = \text{CAM}_g(F_g^{'}) = \mathcal{M}_g^{att} \odot F_g \in \mathbb{R}^{H_g^{'} \text{ x } W_g^{'} \text{ x } C^{'}}. \tag{3.1}$$

Here, the Hadamard product is indicated by the $\odot$ symbol.

Satellite Branch

The street cell attention module creates one mask to reweigh the entire street feature map. However, the satellite cell attention module selects features in the satellite feature map for each vertical cell in the street image. A street image has NM vertical cells because the image is divided horizontally into N slices and vertically into M vertical cell rows. For creating a mask for a specific vertical cell, the satellite feature map and associated cell descriptor are used (see the light gray arrow in Fig. 3.8).

The process of constructing a spatial mask $\mathcal{M}_{s,(m,n)}^{att}$ for a specific vertical cell consists of three steps. Here $m \in \{1,2,..,M\}$ is the index of the vertical cell row, and $n \in \{1,2,..,N\}$ is the index of the street slice. First, the cosine similarity between cell descriptor $f_{g,(m,n)}$ and all satellite features is determined. This results in a matching map with a shape of $H_s^{'}$ x $W_s^{'}$ x 1. Second, the matching map is concatenated to the satellite feature map so that a feature volume is obtained with a shape of $H_s^{'}$ x $W_s^{'}$ x $(C^{'}+1)$. Lastly, 1x1 convolution is used to determine the mask $\mathcal{M}_{s,(m,n)}^{att}$. The concatenation of the matching map is similar to how the street global image descriptor is fused with the satellite features in the architecture of the MCC method [92]. The MCC method is one of our baselines (see Section 3.2).

The satellite cell attention module CAM$_s$ uses the Hadamard product to obtain the reweighed satellite feature map for each vertical cell,

$$F_{s,(m,n)}^{'} = \text{CAM}_s(F_s, f_{g,(m,n)}) = \mathcal{M}_{s,(m,n)}^{att} \odot F_s \in \mathbb{R}^{H_s^{'} \text{ x } W_s^{'} \text{ x } C^{'}}. \tag{3.2}$$

Subsequently, the obtained reweighed feature maps for each vertical cell are concatenated along the feature channel to obtain the final reweighed satellite feature map,

$$F_s^{'} = F_{s,(1,1)}^{'} \oplus ... \oplus F_{s,(m,1)}^{'} \oplus ... \oplus F_{s,(1,n)}^{'} \oplus ... \oplus F_{s,(m,n)}^{'} \in \mathbb{R}^{H_s^{'} \text{ x } W_s^{'} \text{ x } NMC^{'}}. \tag{3.3}$$

Here, the order is based on the vertical cell index m and slice index n. The $\oplus$ symbol indicates concatenation.

3.4.3. Component 3: Slice Descriptor Generator

The slice descriptor generators of the street and the satellite view branch are denoted by SDG$_g$ and SDG$_s$, respectively. The process of generating the slice descriptors consists of 3 steps: (1) determining the cell descriptors, (2) combining the cell descriptors into a slice descriptor, and (3) normalizing the obtained slice descriptor. In metric learning, it is common to normalize descriptors so that they can be compared [44].

Street Branch

A street cell descriptor $f_{g,(m,n)}$ is obtained by averaging the features of the reweighed street feature map $F_g^{'}$ belonging to the specific vertical cell. This is done by computing the Hadamard product spatially between the reweighed feature map and a cell mask $\mathcal{M}^{cell}_{g,(m,n)}$. This mask has the same spatial size as the reweighed feature map, and each weight of the mask indicates whether a feature belongs to the specific cell (weight equal to one) or does not belong to the cell (weight equal to zero). Subsequently, all elements of this product are spatially summed and then divided by the sum of all weights of the mask, i.e., the number of features belonging to the cell.

The street slice descriptor generator SDG$_g$ concatenates all M cell descriptors $f_{g,(m,n)}$ of a slice n to obtain the slice descriptor,

$$\hat{f}_{g,n} = \text{SDG}_g(F_g, n) = f_{g,(1,n)} \oplus ... \oplus f_{g,(m,n)} \in \mathbb{R}^{MC^{'}} \text{ with } \|\hat{f}_{g,n}\|_2 = 1 \text{ for } n \in \{1, 2, .., N\}. \tag{3.4}$$

Here, the order is based on the vertical cell index m, and the obtained slice descriptor is normalized.

Satellite Branch

The satellite slice descriptor generator SDG$_s$ does require a street camera pose $\xi_A = (h_A, w_A, \theta_A)$ for the horizontal slicing since the pose defines the edges of the N slices. A satellite image has slices but no cells. However, a reweighed version of the satellite feature map has been determined for each vertical cell in the street image. Therefore, for each slice, there are M reweighed feature volumes, each with a shape of $H_s^{'}$ x $W_s^{'}$ x $C^{'}$.

The satellite cell descriptor $f_{s,(m,n)}$ is obtained by averaging the features of the reweighed satellite feature map $F^{'}_{s,(m,n)}$ belonging to slice n. This is done by first multiplying the reweighed feature map by a slice mask $\mathcal{M}^{slice}_{g,(m,n)}$. Subsequently, all elements of this product are spatially summed and then divided by the sum of all weights of the mask. So for a satellite cell descriptor, the entire slice is used, while for a street cell descriptor, only a cell of the slice is used.

The satellite slice descriptor generator SDG$_s$ concatenates all M cell descriptors $f_{s,(m,n)}$ of a slice n to obtain the slice descriptor,

$$\hat{f}^A_{s,n} = \text{SDG}_s(F_s, \xi_A, n) = f^A_{s,(1,n)} \oplus ... \oplus f^A_{s,(m,n)} \in \mathbb{R}^{MC^{'}} \text{ with } \|\hat{f}^A_{s,n}\|_2 = 1 \text{ for } n \in \{1, 2, .., N\}. \tag{3.5}$$

Here, the order is again based on the vertical cell index m, and the obtained slice descriptor is normalized.

3.4.4. Image Descriptors

In Subsection 3.3.2, it has been stated that the spatial structure of the horizontal slicing, i.e., which slices are adjacent to a particular slice, can be used for the matching. Only permutations of the satellite slice descriptors that are consistent with the spatial structure of the slicing need to be matched. The slice indices n of the street and satellite slices depend on the street image orientation and the pose used in the satellite image, respectively. By concatenating the slice descriptors according to the order of the slice indices, a descriptor is obtained for each image that incorporates this information.

Street Branch
The street descriptor is obtained by concatenating the N normalized street slice descriptors,

$$\hat{f}_g = \frac{1}{\sqrt{N}}(\hat{f}_{g,1} \oplus ... \oplus \hat{f}_{g,N}) \in \mathbb{R}^{NMC^{'}} \text{ for } n \in \{1, 2, .., N\} \text{ and } \|\hat{f}_g\|_2 = 1. \tag{3.6}$$

Here, the order is based on the slice index n. The $1/\sqrt{N}$ term is for the normalization of the descriptor as there are N slice descriptors, each with unit length. The descriptor consists of NM cell descriptors each with a length $C^{'}$. Thus, the image descriptor has a length $NMC^{'}$.

Satellite Branch
Similarly, the normalized satellite descriptor can be obtained by concatenating the N normalized satellite slice descriptors and multiplying by $1/\sqrt{N}$,

$$\hat{f}_s^A = \frac{1}{\sqrt{N}}(\hat{f}_{s,1}^A \oplus ... \oplus \hat{f}_{s,N}^A) \in \mathbb{R}^{NMC^{'}} \text{ for } n \in \{1, 2, .., N\} \text{ and } \|\hat{f}_s^A\|_2 = 1. \tag{3.7}$$

3.4.5. Similarity

The similarity between two image descriptors is measured using the cosine similarity,

$$S_A = \text{sim}(\hat{f}_g, \hat{f}_s^A) = \hat{f}_g \cdot \hat{f}_s^A \text{ with } -1 \leq S_a \leq 1. \tag{3.8}$$

Here, the cosine similarity is equal to the dot product between the two descriptors because both image descriptors have been normalized.

3.4.6. Loss Function

The training objective is to make the similarity S_{GT} higher than the similarity S_A for all poses $\xi_A \neq \xi_{GT}$. A modified version of the infoNCE loss [52] from contrastive representation learning [32] is used as a loss function,

$$\mathcal{L} = -\log\left(\frac{\exp(S_{GT}/\tau)}{\alpha(\frac{1}{T}\sum_{t=1}^{T}\exp(S_{A_t}/\tau)) + \exp(S_{GT}/\tau)}\right) \tag{3.9}$$

Here, T is the number of (predetermined) negative poses. We introduce hyperparameter α in the loss. The original infoNCE loss can be obtained by taking $\alpha = T$. The infoNCE loss can be seen as a generalized version of the triplet loss [67] and is used in image retrieval in the case multiple negative samples are presented at the same time.

For each image pair, T negative poses ξ_{A_t} are used and the positive, i.e., ground truth, pose ξ_{GT}. The adjustment of the infoNCE loss by introducing hyperparameter α concerns the scaling of the negative term in the denominator. The average negative term is computed by dividing by the number of negatives T. This average is scaled by the term α. The hyperparameter τ has a similar role as the margin between the positive and negative samples in the triplet loss [92].

3.4.7. Street Camera Pose prediction

The pose of the street camera can be predicted by determining for *K* poses ξ_k the similarity S_k and taking the pose ξ_k with the highest similarity S_k,

$$\xi_{predicted} = \xi_k \text{ with } k = \text{argmax}_{S_k}(\{S_k \mid k \in \{1, ..., K\}\}). \tag{3.10}$$

Here, the accuracy with which the true street camera pose ξ_{GT} can be approximated depends on the number of poses used. For pose estimation, $(H_s/\Delta r+1)(W_s/\Delta r+1)(360°/\Delta\theta)$ poses are required if for localization and orientation estimation a resolution of Δr (pixels) and $\Delta\theta$ (degrees) is used, respectively. When the orientation of the street image is known, pose estimation becomes localizing, and then there are $(H_s/\Delta r + 1)(W_s/\Delta r + 1)$ poses required. This comes down to the situation where there are K locations used, and at each location, the true orientation is used for the slicing in the satellite image.

3.5. SliceNet Architecture compared to Baselines

SliceNet, similar to the CVR model [107] and the MCC model [92], uses a VGG16 [76] as a feature extractor for both branches. Furthermore, all three methods process the feature maps and fuse the information from the two branches to determine the camera location. However, the way the features are used to determine the street camera location in the satellite image differs.

The CVR model uses a SAFA module [73] as the descriptor network and creates a global image descriptor for both images. These descriptors are concatenated, and an MLP is used to predict the camera location. The MCC model also uses SAFA modules. However, it divides the satellite image into a grid and determines a descriptor for each grid cell. Then the street global image descriptor is matched with the satellite descriptors to obtain a matching map. The decoder uses this matching map, the satellite descriptors, and the satellite feature map to predict a probability distribution for the camera location. SliceNet, on the other hand, divides both images into N slices and creates a slice descriptor for each slice. For matching, the slice descriptors of an image are concatenated, and the obtained image descriptors are matched using the cosine similarity.

It can be noted that there are two crucial differences between SliceNet and baselines. First, SliceNet creates more than one descriptor for both images, while the baselines create a single global image descriptor for the street image. Second, SliceNet indirectly predicts the camera location by taking the argmax over the similarities of different poses, while the baselines directly predict the location.

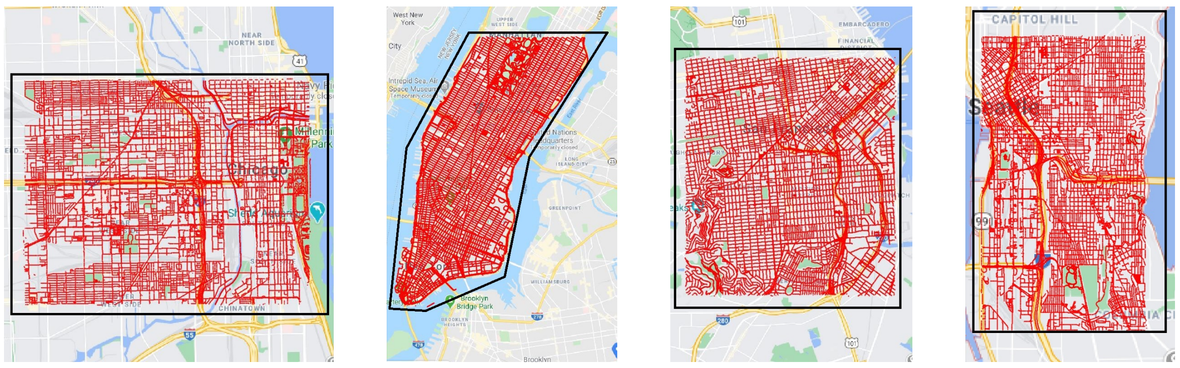
4 Experiments

In this chapter, we first introduce the dataset in Section 4.1 and the evaluation metrics for our experiments in Section 4.2. Then, we provide the implementation details of SliceNet in Section 4.3. Subsequently, Section 4.4 and Section 4.5 present the proof of concept for SliceNet and the ablation studies, respectively. In Section 4.6, SliceNet is compared to the two state-of-the-art baselines to show our advantage in metric localization in generalizing to new measurements in the same area and across areas. Finally, in Section 4.7, the runtime performance of SliceNet is compared to the baselines.

4.1. Dataset

The VIGOR [107] dataset is used to evaluate the performance of SliceNet. This dataset contains street panorama images. Due to their unlimited horizontal field of view, these images can more clearly show the advantage of using the azimuth prior for learning local correspondences between the two views than street images with a limited field of view. Besides, the VIGOR dataset allows for both same-area and cross-area evaluation (see Tab. 2.2).

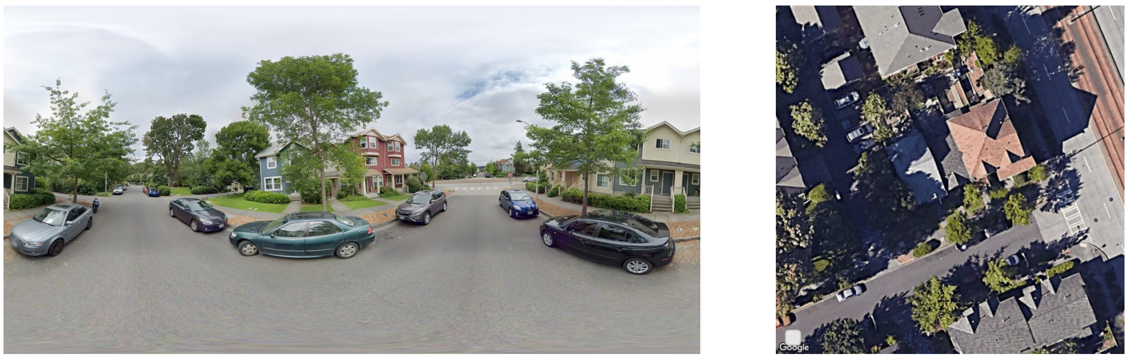
The VIGOR dataset contains geo-tagged street and satellite image pairs collected in four cities in the United States, namely (1) Chicago, (2) New York, (3) San Francisco, and (4) Seattle. Fig. 4.1 shows the distribution of the street images and the aerial coverage of the satellite images. The raw image sizes for the street view and the satellite view are 2048 x 1024 pixels and 640 x 640 pixels, respectively.



(a) Chicago **(b)** New York **(c)** San Francisco **(d)** Seattle

Figure 4.1: The distribution of the street images (red dots) and the aerial coverage of the satellite images (black polygon) of the VIGOR dataset. The images are taken from the work of Zhu et al. [107].

Industrial-grade GPS tags for both street and satellite images are provided for meter-level evaluation [107]. The street image's geo-tag indicates the street camera's location, while the satellite image's geo-tag indicates the satellite image's center. The street images are 360° panorama images. Besides, the image pairs are aligned so that the vertical centerline of the street image corresponds to North in the satellite image, i.e., the orientation of a street image with respect to the satellite image is equal to zero. Fig. 4.2 shows an example of a street and satellite image pair from the dataset.



(a) Street (360° panorama) image **(b)** Satellite image

Figure 4.2: A street and satellite image pair from the VIGOR dataset [107]. The image pair shows a location in Seattle.

As mentioned in the introduction (see Chapter 1), the localization of the autonomous vehicle involves two steps. First, a localization prior is used to obtain a coarse localization. Subsequently, street-to-satellite image metric localization can be used to refine the coarse localization. The required satellite image is extracted from the satellite map using the coarse localization. The required size of the area visible in the satellite image depends on the accuracy of the coarse localization.

The image pairs of the VIGOR dataset are categorized into *positive* and *semi-positive* image pairs. An image pair is denoted as *positive* if the street image's location is within the satellite's center quarter area. Otherwise, it is *semi-positive*. As a result, the coarse localization must have a maximum error of about 18 meters for the positive image pairs and about 36 meters for the semi-positive image pairs. As has been stated by Reid et al. [58], the localization upper bound on local streets for autonomous vehicles is 0.29 meters, and the required update frequency is at least 100 Hertz. Suppose the system achieves this performance. In that case, the previous fine localization can be used as a coarse localization estimate for the current location, and the error is significantly smaller than 18 meters. Therefore, only the positive image pairs are used in this work.

4.1.1. Corrected Ground Truth Labels

The original ground truth locations of the dataset contain an error due to an incorrect ground resolution that had been used to convert a geo-location to an image location. Therefore, we have determined a new ground resolution for each city, and the labels of the dataset have been corrected. More information about the corrected ground truth labels can be found in Appendix A. The original *positive* split with the new ground truth labels has been used for the experiments.

4.1.2. Data Splits

We adopt the *same-area* and *cross-area* splits from Zhu et al. [107] to test the model's generalization in the same cities and across different cities. The same-area setting uses image pairs from all four cities for the training and testing, while the cross-area setting only uses images from New York and Seattle for training and Chicago and San Francisco for testing. This split corresponds to a training and testing dataset size of 52609 and 52605 image pairs for same-area and 51520 and 53694 image pars for cross-area. Tab. 4.1 shows the number of image pairs for the four different cities for the same-area and cross-area settings. We use the same-area training image pairs of Chicago as *tuning split* for hyperparameter optimization and ablation studies.

Table 4.1: The number of positive image pairs for the same-area and cross-area settings of the VIGOR dataset [107].

	Same-area		Cross-area	
	Training	Testing	Training	Testing
Chicago	12740	12739	0	25479
New York	13885	13884	27769	0
San Francisco	14108	14107	0	28215
Seattle	11876	11875	23751	0

4.2. Evaluation Metrics

In Subsection 3.4.7, it is described that SliceNet determines for *K* poses ξ_k the similarity S_k. The pose with the highest similarity is used as the prediction for the camera location.

Localization Performance

The localization error is computed for each image pair to measure the localization performance. This error is given by,

$$l_{error} = \rho_{ground} \cdot \|X_{GT} - X_{pred}\|_2. \tag{4.1}$$

Here, $X_{GT} = (h_{GT}, w_{GT})$ and $X_{pred} = (h_{pred}, w_{pred})$ are the ground truth and predicted street camera location in the satellite image in meters, respectively. The variable ρ_{ground} is the ground resolution of the satellite image in meters/pixel. We follow the convention of Xia et al. [92] and report the mean and median localization error over all test image pairs as localization performance.

The mean localization error can be sensitive to outliers. Sometimes a satellite image has multiple locations similar to the local surroundings visible on the street image. However, only the location with the highest similarity is used as the prediction. Using a visually similar location as a prediction can result in a relatively large error, and, as a result, these large errors may influence the mean error significantly. The median error is more robust against outliers because the value of the midpoint of the error distribution is taken.

Orientation Estimation Performance

If the street camera orientation is unknown, the orientation of the predicted pose can be used as an estimate for the orientation. For measuring the orientation estimation performance, the absolute orientation error is used. This error is given by,

$$\theta_{error} = \text{min}(\theta_{GT} - \theta_{pred},\ 360^\circ - (\theta_{GT} - \theta_{pred})) \in [0, 180^\circ]. \tag{4.2}$$

Here, θ_{GT} and θ_{pred} are the ground truth and predicted orientation of the street camera in the satellite image in degrees, respectively. As for the localization error, the mean and median error over all test image pairs are reported.

4.3. Implementation Details

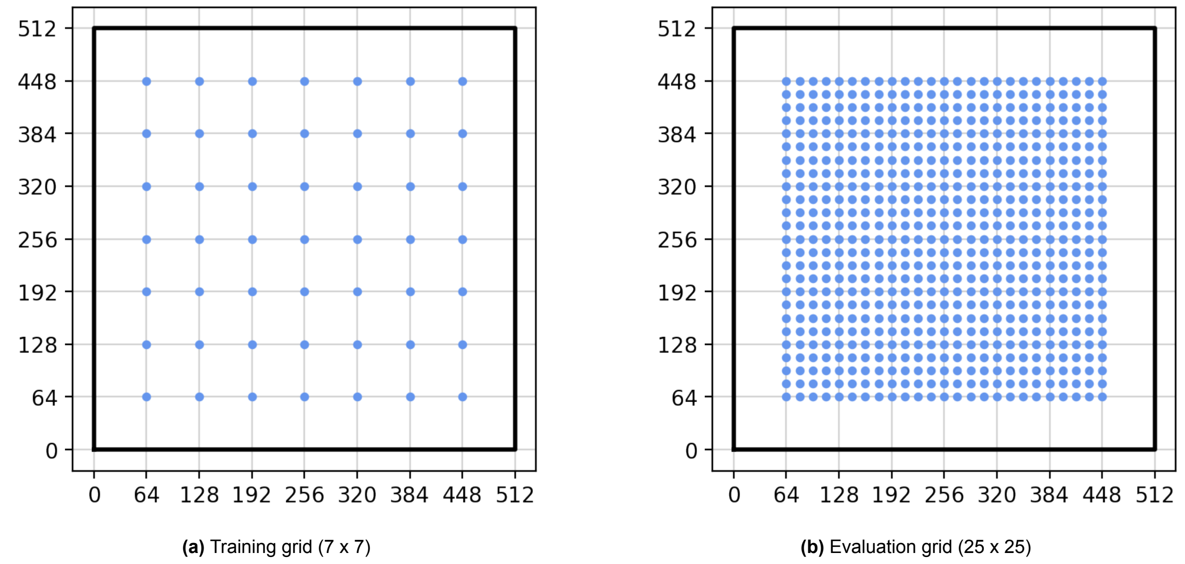
For SliceNet Basic and SliceNet Select, VGG16 [76] up to stage 5 is used as the feature extractor for both branches, and the activation function and pooling operation of the last layer are removed. The feature extractors do not share their weights, and each feature extractor's stride is equal to 16. The used VGG feature extractors have been pre-trained on ImageNet [22].

Each cell attention module uses two sequential convolution layers with a kernel size of 1 to create a spatial mask given a feature volume. The first convolution layer has 64 kernels, and the number of kernels of the second layer equals the number of masks the cell attention module needs to create. For the street view, this is 1 mask, and for the satellite view, these are NM masks (see Subsection 3.4.2). Before every convolutional layer, batch normalization in combination with a ReLU activation function is used. After the second convolutional layer, a Sigmoid activation function is used to obtain the mask.

Each slice descriptor generator has the number of horizontal slices N and the number of vertical cells M as hyperparameters, and it has no learnable parameters. We initially set the number of horizontal slices to eight for each view and present an ablation study about the effect of the number of horizontal slices in Subsection 4.5.1. Only SliceNet Select can use more than one vertical cell per slice because the attention masks are needed to select different features in the satellite image. So for SliceNet Basic, one vertical cell is used. For SliceNet Select, we initially set the number of vertical cells to one, and we present an ablation study about the effect of the number of vertical cells in Subsection 4.5.2.

Predetermined poses for loss function

The loss function (see Section 3.4) uses T negative poses and 1 positive pose, i.e., the ground truth pose. A hyperparameter search was done to determine the appropriate number of grid locations for training. The investigation shows that a better localization performance can be obtained if more grid locations are used. However, the performance increase stagnates if grids are used with more than 7 locations per dimension. Therefore it has been decided to use a uniform grid of 7 x 7 locations during



(a) Training grid (7 x 7) **(b)** Evaluation grid (25 x 25)

Figure 4.3: The location grid used for the training and evaluation of SliceNet. The black contour indicates the edges of the satellite image. The resized satellite image has a resolution of 512 x 512 pixels. (a) The location grid for training. (b) The location grid for evaluation.

the training. On the other hand, a larger grid is used for evaluation so that the street camera pose can be accurately determined: a uniform grid of 25 x 25. Fig. 4.3 shows the grids used for the training and evaluation.

Optimizing model parameters
For the loss function, a hyperparameter search was done with $\alpha \in \{2, 4, 8, 16\}$, and based on those results, $\alpha = 4$ was taken. Furthermore, $\tau = 0.1$ is used as was done by Oord et al. [52] and Xia et al. [92]. The loss has been optimized by the Adam optimizer [33] with a learning rate of 10^{-5}. For this optimizer, the default values $\beta_1 = 0.9$ and $\beta_2 = 0.999$ were used. A batch size equal to four was used for each experiment unless otherwise stated.

Input images and feature map sizes
The street and satellite images are resized to 320 x 640 pixels and 512 x 512 pixels following the convention of Xia et al. [92]. The stride of the feature extractor is 16, and the last convolutional layer of the feature extractor has 512 kernels. As a result, the street and satellite feature maps have a shape of 20 x 40 x 512 and 32 x 32 x 512, respectively. The reweighed feature maps have the same spatial size as the original feature maps.

4.4. Proof of Concept

In Section 3.3, we hypothesized whether we can learn orientation discriminative descriptors for the street and satellite slices where descriptors of matching slices are similar, i.e., close together in the feature space of the descriptors, and descriptors of non-matching slices are dissimilar. We refer to this as the *orientation hypothesis*. The ground truth street camera pose is required for the horizontal slicing of the satellite view. So the orientation discriminative descriptors could only be used to determine which street slice belongs to which satellite slice. However, the slices cannot be made during inference as this requires the ground truth street camera pose.

In order to reduce the dependence on the camera pose, we hypothesized that we can learn location discriminative satellite slice descriptors for which the satellite slice descriptors belonging to the street camera location are more similar to the street slice descriptors than the satellite descriptors of other locations in the satellite image. For these location discriminative descriptors, only the ground truth orientation is needed during inference. So if the street and satellite image are aligned, then the slicing can be done, and the camera location can be predicted. This hypothesis is referred to as the location hypothesis.

We further stated that the concepts for orientation and location estimation can be combined to determine the location and orientation of unaligned street images. During training, the ground truth pose is needed, and during inference, the camera pose can then be predicted by measuring the similarity for different poses in the satellite image. This is referred to as the *pose* hypothesis. We hypothesize whether we can learn pose discriminative satellite slice descriptors for which the satellite slice descriptors belonging to the street camera pose are more similar to the street slice descriptors than the satellite descriptors of other poses in the satellite image.

For testing the hypotheses, a proof of concept experiment is designed using SliceNet Basic to determine the camera location for aligned image pairs and the camera location and orientation for unaligned image pairs. Three different versions of the loss function (see Section 3.4) are used to test the hypotheses, i.e., one loss version for each hypothesis. These versions are referred to as the *orientation loss*, *location loss*, and *pose loss*. The difference between the loss versions is the negative poses that are used.

Positive pose

The street feature map has a spatial size of 20 x 40 features and is divided with horizontal slicing into 8 slices of 20 x 5 features each. The slice descriptor of each street slice is obtained by averaging the features of the slice. The ground truth street camera pose can then be used to divide the satellite feature map into eight slices too. These satellite slices can have different sizes depending on the camera location in the satellite image. Similarly to the street view, the satellite slice descriptors can be obtained by averaging the features of each slice. In this manner, eight street slice descriptors are obtained for the street view, and eight satellite slice descriptors are obtained for the satellite view. The slice indices are used to combine the slice descriptors into a single descriptor for each image. This results in a street descriptor and satellite descriptor, each with a length of 4096, i.e., eight slice descriptors with a length of 512. This satellite descriptor is used to determine the positive similarity in the loss function because this descriptor is based on the ground truth camera pose.

Negative poses

During training, the slices of the street and satellite image must be aligned such that each street slice overlaps with only one satellite slice so that there is one matching satellite slice for each street slice. This results in two options for the negative poses that can be used: (1) for each location, the ground truth orientation can be used, and (2) for each location, the N possible orientations can be used. In the case of the first option, it is assumed that the orientation of the street image is known, while for the second option, it is assumed that each slice has one matching satellite slice and that the orientation is one of the N possibilities.

For the orientation hypothesis, the second option is used in combination with the ground truth location. This results in seven negative poses since one of the eight orientations is the true orientation. The training grid is used for the location and pose hypotheses, i.e., 7 x 7 grid locations. All grid locations are considered negative locations even though the ground location may be close to one of the grid locations. Such grid locations are regarded as hard negative locations. The location hypothesis uses the ground truth orientation, so 49 negative poses are used, i.e., one pose for each grid location. In contrast, the pose hypothesis uses all eight orientations for each location. This results in eight poses per grid location and, thus, 392 negative poses.

Experiment

The three loss versions are used to train instances of SliceNet Basic. For training and evaluation SliceNet, the tuning split is divided into a training and validation dataset with an 80/20 split. Evaluation is done with aligned and unaligned image pairs. Aligned image pairs are used to quantify the localization performance. Since these image pairs are aligned, slicing can be done during inference, where each slice has one matching slice in the other view. In contrast, unaligned image pairs are used to measure localization and orientation performance. This means that each street image has a random orientation which may cause the slices to be unaligned. This differs from the training objective since, during training, each slice only overlaps with one slice in the other view. However, the purpose of this experiment is only to test whether the learned descriptors are indeed orientation, location, or pose discriminative. Each configuration is trained three times, and the instance with the lowest mean validation localization error for aligned image pairs is used.

Table 4.2: Localization error for aligned and unaligned image pairs on the tuning split of the VIGOR dataset [107]. The orientation error is also listed for unaligned image pairs. Best performance in **bold**. *Coarse localization* denotes using the satellite image center as the prediction and North as orientation. SliceNet Basic is denoted by *SB*.

	Aligned image pairs		Unaligned image pairs			
	Localization error (m)		Localization error (m)		Orientation error (deg)	
	Mean	Median	Mean	Median	Mean	Median
Coarse localization	14.40	15.36	14.40	15.36	90.48	89.44
SB: Orientation loss	10.55	7.61	15.44	11.57	42.53	15.75
SB: Location loss	8.70	**5.98**	11.93	9.02	50.40	18.00
SB: Pose loss	**8.23**	6.41	**11.22**	**8.74**	**39.50**	**15.19**

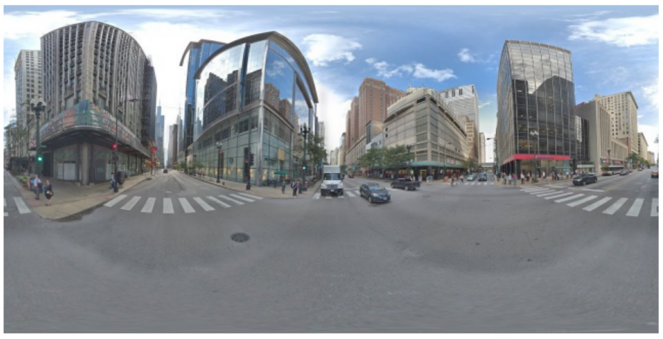
Quantitative Results

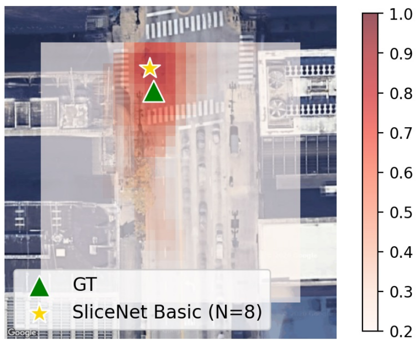
The quantitative comparison between the three losses is summarized in Tab. 4.2. The performance is also given for the *coarse localization*. The coarse localization, e.g., GNSS-based localization, is used to obtain the satellite image and corresponds to the center location of the satellite image. The idea is that street-to-satellite metric localization can improve this coarse localization. Thus, the coarse localization is considered the baseline for the proof of concept. For the orientation estimation, North is used as the baseline. It was expected that with North as the orientation estimate, a mean orientation error would be obtained of about 90°. The minimum and maximum orientation errors are 0° and 180°, respectively. Thus, 90° is the midpoint of this range. This expectation is consistent with the results since the mean and median orientation error obtained using North as prediction are both approximately equal to 90°.

The SliceNet Basic instances have been optimized for the mean error, and the pose loss is the best-performing loss function for both the aligned and unaligned image pairs. This result is in line with the expectation that the generalization is better when we use more different poses, i.e., more negative slices, during training. The training is done with aligned image pairs, thus not optimized for street images with a random orientation. For all versions of the loss function, the localization errors are significantly higher than those for aligned image pairs. However, this is also expected because unaligned image pairs are more challenging because a street slice may overlap with two satellite slices. Interestingly, training with the location loss makes the orientation estimation capability of SliceNet Basic close to the orientation estimation capability of SliceNet Basic trained with the orientation loss.

Qualitative Results

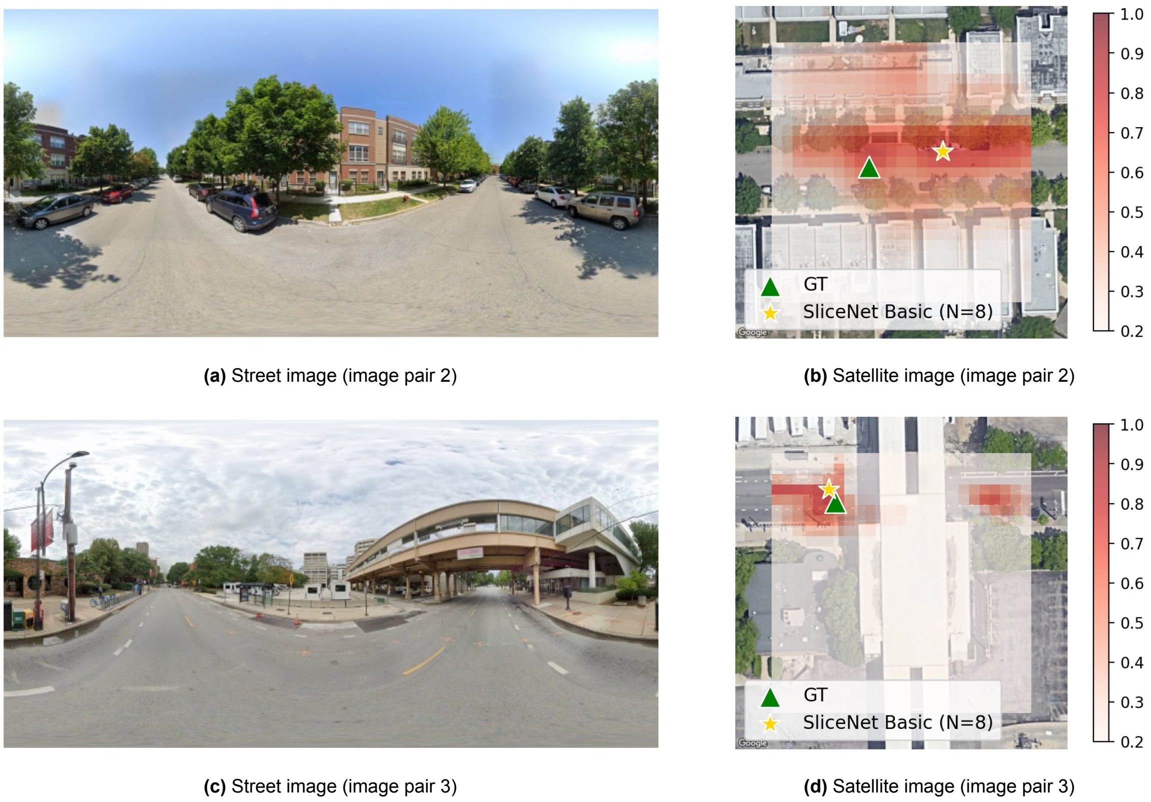
Fig. 4.4 and Fig. 4.5 show three aligned image pairs with the similarity map that has been created with SliceNet Basic (*pose loss*). It can be seen that the predicted location is close to the ground truth location for the first and third image pair. However, Fig. 4.5b shows a relatively large region of uncertainty, and the prediction is relatively far from the ground truth location. In general, the similarity is low for locations located on the roof of a building, and high regions of similarity are aligned with the road. Lastly, Fig. 4.5d shows a multi-modal output.


(a) Street image (image pair 1)



(b) Satellite image (image pair 1)

Figure 4.4: Image pair 1 with the corresponding similarity map in overlay on the satellite image. SliceNet Basic (*pose loss*) with eight horizontal slices has been used. The image pair is aligned. The images are part of the test split of Chicago (same-area) of the VIGOR dataset [107].



(a) Street image (image pair 2) **(b)** Satellite image (image pair 2)

(c) Street image (image pair 3) **(d)** Satellite image (image pair 3)

Figure 4.5: Image pairs 2 and 3 with the corresponding similarity map in overlay on the satellite image. SliceNet Basic (*pose loss*) with eight horizontal slices has been used. All image pairs are aligned. The images are part of the test split of Chicago (same-area) of the VIGOR dataset [107].

Preliminary conclusions

First, SliceNet Basic achieves significantly better mean and median orientation error for unaligned image pairs with the orientation loss than using North as prediction. Therefore, it is concluded that SliceNet Basic trained with the orientation loss learned orientation discriminative slice descriptors. Second, the SliceNet Basic instance trained with the location loss achieves a considerably better mean and median localization performance for the aligned image pairs than the coarse localization. Moreover, a better mean and median localization performance are also obtained for the unaligned image pairs. Therefore, it is concluded that the horizontal slices can be used to learn location discriminative descriptors. Lastly, the localization and orientation error of SliceNet Basic trained with the pose loss for unaligned image pairs is significantly better than the localization performance obtained with the coarse localization and North as prediction. Therefore, we conclude that the horizontal slices can be used to learn pose discriminative slice descriptors. In the other experiments, the pose loss will be used to train SliceNet, and we will only use aligned image pairs, i.e., we do localization only.

4.5. Ablation Study

Section 4.4 presented the proof of concept for SliceNet. The results show that SliceNet Basic can achieve better localization performance than the coarse localization, i.e., the center of the satellite image. For that experiment, eight horizontal slices were used for both views. In Subsection 4.5.1, the effect of the number of horizontal slices on localization performance is studied.

SliceNet Select can use the cell attention modules to select features in the street and satellite feature maps. In this regard, a street slice can be divided into M vertical cells. Subsection 4.5.2 presents an ablation study on the effect of the number of vertical cells on the localization performance and the effect of using a street cell descriptor to create the mask.

4.5.1. Number of Horizontal Slices

This subsection describes the experiment for studying the effect of the number of horizontal slices on the localization performance. The experiment is explained first, and then the quantitative and qualitative results are presented. Finally, some conclusions are drawn based on the results.

Experiment
In order to study the effect of the number of horizontal slices on localization performance, instances of SliceNet Basic are trained with different values for the number of horizontal slices. Experiments are done with 4, 8, 12, 16, 20, 24, 28, 32, 36, and 40 horizontal slices. The same training and validation split of the tuning split is used as for the proof of concept experiment. It is assumed that image pairs are aligned, i.e., the ground truth orientation can be used for creating the slices during inference. Each configuration is trained three times, and the instance with the lowest mean validation localization error is used. We do this because it has been noticed that the performance of SliceNet sometimes varies for the same experiment. This way, we try not to let this variation affect the results.

Quantitative results
The quantitative comparison between the different numbers of horizontal slices is summarized in Tab. 4.2. The performance of the coarse localization and SliceNet with 8 horizontal slices are from the proof of concept experiment. In general, a lower mean and median error can be obtained by using more horizontal slices. This is in line with the expectation that more horizontal slices enable a more accurate localization because when more horizontal slices are used, a small change of the location reduces the overlap between matching slices more. However, it can be seen that the increase in performance saturates. The mean localization error for 8 horizontal slices is remarkably low compared to the performance of 4, 12, 16, and 20 horizontal slices.

Table 4.3: Localization error for different number of horizontal slices on the tuning split of the VIGOR dataset [107]. Best performance in **bold**. *Coarse* denotes the coarse localization, i.e., using the satellite view image center as the prediction.

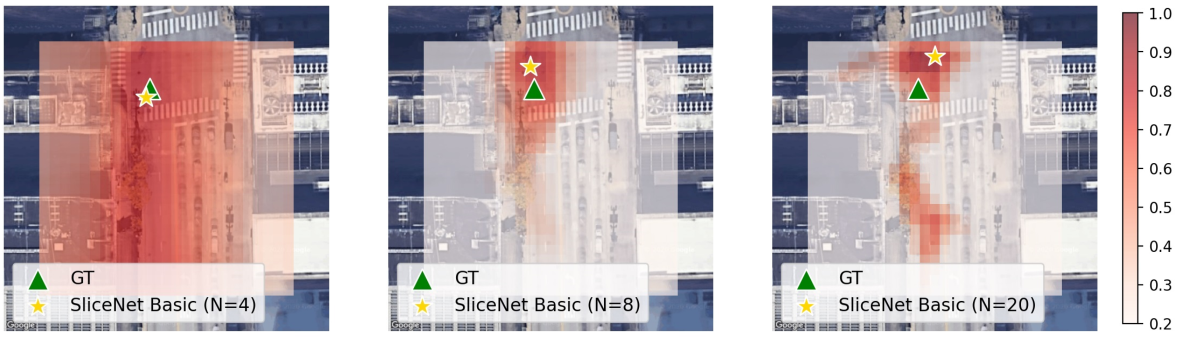
		Number of horizontal slices (N)									
	Coarse	4	8	12	16	20	24	28	32	36	40
Mean	14.40	9.30	8.23	9.13	8.89	8.51	8.41	8.28	8.63	**8.18**	8.68
Median	15.36	7.04	6.41	6.74	6.34	5.73	5.78	**5.38**	5.62	5.66	5.62

Qualitative results
Fig. 4.6 shows the similarity map for three different satellite images for 4, 8, and 20 horizontal slices. The same three image pairs were used as for the proof of concept of SliceNet. It can be seen that there are large differences between the similarity maps. The similarity maps of SliceNet with 4 horizontal slices show relatively high uncertainty. In those similarity maps, almost every grid location has the same similarity. However, for all three image pairs, the predicted location is close to the ground truth location. The similarity maps of SliceNet with 8 and 20 horizontal slices show considerably less uncertainty than those of SliceNet with 4 horizontal slices. Moreover, the similarity maps for image pair 3 show a clear multi-modal distribution. Besides, it can be seen that the uncertainty in the similarity maps decreases for an increasing number of horizontal slices. This is consistent with the quantitative results since they show the trend that the mean and median localization error decreases if more horizontal slices are used.

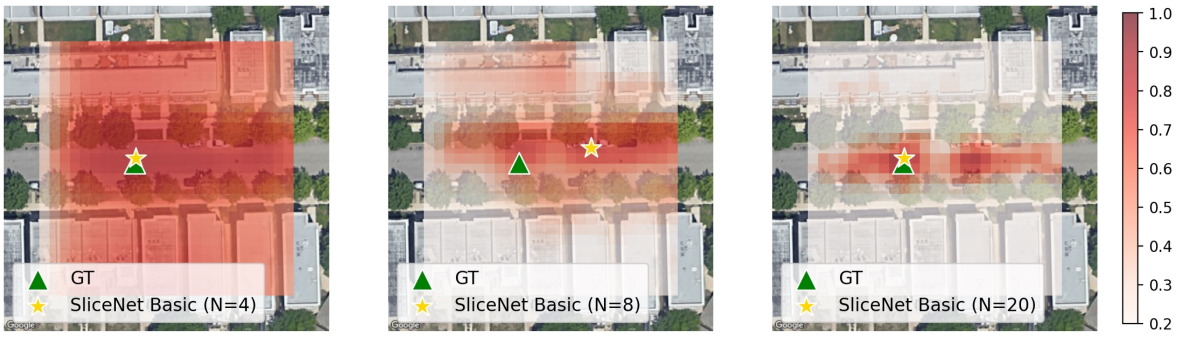
Preliminary conclusions
It can be concluded that both the mean and median localization error decrease if more horizontal slices are used. Thus, a higher localization performance can be achieved with more horizontal slices. In addition, the qualitative results show that the similarity maps show less uncertainty if more horizontal slices are used. Both the quantitative and the qualitative results are in line with the expectation that the localization performance improves when more horizontal slices are used because when more horizontal slices are used, a small change of the location reduces the overlap between matching slices more. However, the increase in localization performance does plateau when more than about 20 horizontal slices are used. Therefore, we will use 20 horizontal slices for our best model.

4.5.2. Number of Vertical Cells

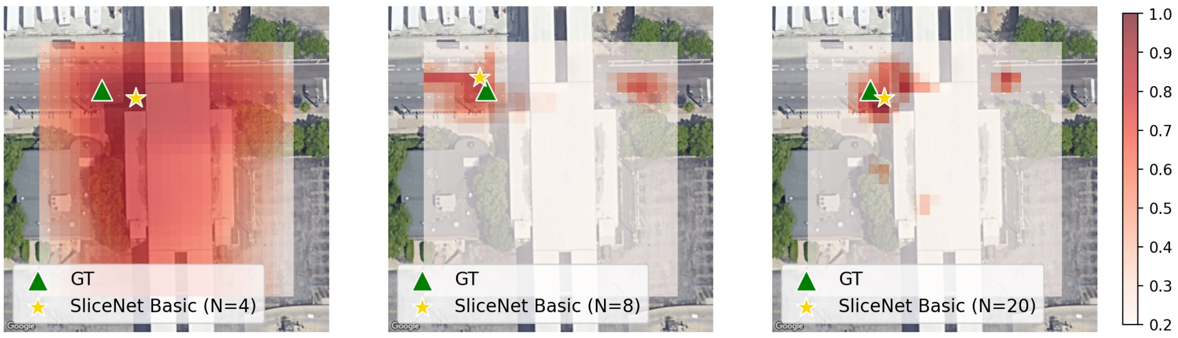
In Section 3.3, we hypothesized whether we can learn to select objects in the satellite image that correspond to a specific vertical cell row and make discriminative cell descriptors that can be used for localizing the street camera in the satellite image. Besides, we noted that a street cell descriptor could be used to search for a specific cell instead of an entire vertical cell row. Therefore, we hypothesized



(a) $N = 4$ (image pair 1) **(b)** $N = 8$ (image pair 1) **(c)** $N = 20$ (image pair 1)



(d) $N = 4$ (image pair 2) **(e)** $N = 8$ (image pair 2) **(f)** $N = 20$ (image pair 2)



(g) $N = 4$ (image pair 3) **(h)** $N = 8$ (image pair 3) **(i)** $N = 20$ (image pair 3)

Figure 4.6: The similarity maps for three different values for the number of horizontal slices: 4, 8, and 20. SliceNet Basic (*pose loss*) has been used. The corresponding street image for each satellite image can be seen in Fig. 4.4 and Fig. 4.5. The images are part of the test split of Chicago (same-area) of the VIGOR dataset [107].

whether we can learn to select objects in the satellite image that correspond to a specific vertical cell of a street slice and make discriminative cell descriptors that can be used for localizing the street image in the satellite image.

SliceNet Select reweighs the feature maps that follow from the feature extractors and divides the column of the street image into M parts (see Section 3.4). The street cell descriptors can be used to create the attention maps for reweighing the satellite features (see the light gray arrow in Fig. 3.8). For this purpose, the matching map obtained by determining the cosine similarity between a street cell descriptor and the satellite features is used in combination with the satellite feature map to create the attention map. An ablation study has been conducted to investigate the effect of the number of vertical cells and the use of the matching map. Two versions of SliceNet Select were investigated, namely (1) without the matching map, i.e., SliceNet Select (w/o matching map), and (2) with the matching map, i.e., SliceNet Select (matching map). Below, the experiment is first explained, and then the quantitative and qualitative results are presented. Finally, some conclusions are drawn based on the results.

Experiment
In order to study the effect of the number of vertical cells on localization performance, instances of SliceNet Basic are trained with different values for the number of vertical cells. Eight horizontal slices are used. Experiments are done with 1, 2, 4, and 8 vertical cells and using the matching map. The same training and validation split of the tuning split is used as for the proof of concept. It is assumed that image pairs are aligned, i.e., the ground truth orientation can be used for creating the slices during inference. Each configuration is trained three times, and the instance with the lowest mean validation localization error is used.

Quantitative results
The quantitative comparison between the different numbers of vertical slices without the use of the matching map is summarized in Tab. 4.2. The performance of the *coarse localization* and *no attention*, i.e., SliceNet Basic with eight horizontal slices, are from the proof of concept experiment. SliceNet Select (w/o matching map) performs better than SliceNet Basic for all values of the number of vertical cells. Reweighing the feature maps does improve the performance considerably. The best performance is obtained using only one vertical cell, and the performance decreases when the number of vertical cells increases.

In Tab. 4.5, the quantitative comparison between the different numbers of vertical slices with the use of the matching map is summarized. The results for SliceNet Select (matching map) show a similar trend for the number of vertical cells, namely that the performance decreases when more vertical cells are used. SliceNet Select (matching map) also performs better than SliceNet Basic for all used values for the number of vertical cells. It follows from the results of Tab. 4.4 and Tab. 4.5 that using the matching map is beneficial for the performance of SliceNet Select because using the matching map increases the performance for values that have been used for the number of vertical cells.

Table 4.4: Localization error for different number of vertical cells for SliceNet Select (w/o matching map) on the tuning split of the VIGOR dataset [107]. Best performance in **bold**. *Coarse* denotes the coarse localization, i.e., using the satellite view image center as the prediction. *No attention* means not reweighing the feature maps, and this is the performance of SliceNet Basic from Tab. 4.2.

			Number of vertical cells (M)			
	Center-only	No attention	1	2	4	8
Mean	14.40	8.23	**7.01**	7.66	8.14	8.30
Median	15.36	6.41	**4.92**	5.48	6.03	6.20

Table 4.5: Localization error for different number of vertical cells for SliceNet Select (matching map) on the tuning split of the VIGOR dataset [107]. Best performance in **bold**. *Coarse* denotes the coarse localization, i.e., using the satellite view image center as the prediction. *No attention* means not reweighing the feature maps, and this is the performance of SliceNet Basic from Tab. 4.2.

			Number of vertical cells (M)			
	Center-only	No attention	1	2	4	8
Mean	14.40	8.23	**6.70**	6.76	6.86	7.16
Median	15.36	6.41	**4.78**	4.94	4.96	5.29

Qualitative results
Fig. 4.7 shows the similarity map and the attention maps for two image pairs created with SliceNet Select (w/o matching map) with one vertical cell. The similarity maps of SliceNet Select (w/o matching map) are similar to those of SliceNet Basic: the prediction is close to the ground truth location for both image pairs, and the similarity map is aligned with the road. Attention is used in SliceNet Select to reweigh the features of the street and satellite feature map. The feature maps have a smaller spatial size than the input images due to the stride of max-pooling layers of the feature extractor. However, the attention maps can be overlaid on top of the images. The stride of the feature extractor is 16, so each cell from an attention map corresponds to a pixel area of 16 x 16 pixels. Fig. 4.7 shows that SliceNet

Select (w/o matching map) reweighs features that show the horizon near a road more heavily. A visible horizon near a road indicates that the road is heading in that direction. This can be an important cue for creating a slice descriptor. When only one vertical cell is used, the network has the freedom to reweigh features from the entire height of the street feature map and use them to create the slice descriptor. This means that it can be learned to give road and sky a relatively low weight and the horizon near a road a relatively high weight, as can be seen in Fig. 4.7.

SliceNet Select (w/o matching map) with four vertical cells cannot ignore the road, and the sky as this model is forced to create a quarter of the slice descriptor for each vertical cell. In Fig. 4.8g and 4.8h, it can be seen that the model learns to weigh everything the same for vertical cell 4, i.e., all features in that vertical cell are equally discriminative. Besides, in Fig. 4.8c and Fig. 4.8d, it can be seen that SliceNet Select learns to weigh trees for vertical cell 2 more heavily in both the street image and the satellite image. The same can be seen for the road in vertical cell 3 in Fig. 4.8e and Fig. 4.8f. In the street image, the horizon near a road is weighed more heavily, and in the satellite image, the entire road is weighed more heavily.

In general, vertical cells contain different content. Fig. 4.8 shows that vertical cell 2 mainly shows trees and buildings, while vertical cell 3 shows roads and cars (transient objects). It follows from the street and satellite attention maps of this figure that the model has learned to reweigh trees more heavily for vertical cell 2 (see Fig. 4.8c and Fig. 4.8d) and roads for vertical cell 3 (see Fig. 4.8e and Fig. 4.8f). So the model can learn that some vertical cells differ in content and can reweigh the features belonging to that vertical cell more heavily in the satellite view.

Fig. 4.9 shows the similarity map and some attention maps for one image pairs that have been created with SliceNet Select (matching map) with one vertical cell. It can be concluded that SliceNet Select (matching map) shows similar similarity and attention maps as SliceNet Select (w/o matching map). This version of SliceNet uses street cell descriptors to create satellite attention maps. It can be seen that the attention maps for different street slice indices differ considerably. For example, the attention map for street slice 2 mainly pays attention to the horizon near the road in the street image (see Fig. 4.9c) and to the entire road in the satellite image (see Fig. 4.9d). In contrast, for street slice 8, the main focus is on the trees in both the street image and the satellite image (see Fig. 4.9g and Fig. 4.9h).

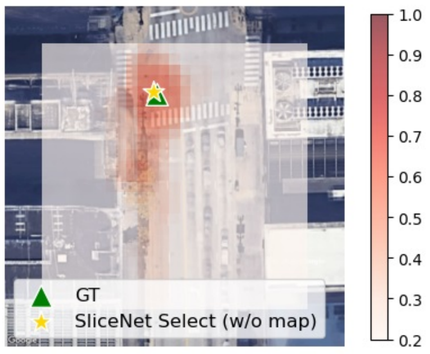
Fig. 4.10 shows the similarity map and some attention maps for one image pair created with SliceNet Select (matching map) with four vertical cells. It can be seen that the attention map for the satellite image differs for each street image cell used. The cell in Fig. 4.10c shows the side of a building, and in the corresponding attention mask of Fig. 4.10d, it can be seen that attention is mainly paid to the sidewalk. Again, this version of SliceNet learns to ignore the sky. Fig. 4.10e and Fig. 4.10f show that sky features are almost uniformly weighed with a relatively low weight compared to the weights in the other attention masks.

Preliminary conclusions

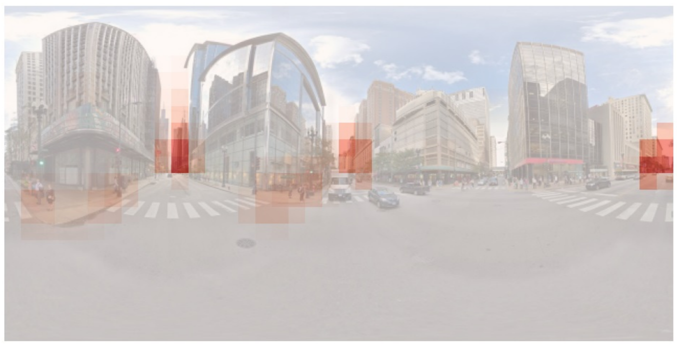
It can be concluded that using cell attention modules can improve the localization performance of SliceNet. For all values used for the number of vertical cells, the performance of SliceNet Select is higher than that of SliceNet Basic. The best performance is achieved with one vertical cell. Besides, the performance monotonically decreases if more vertical cells are used. Finally, using the matching map improves localization performance.

In the created attention maps, it can be seen that SliceNet Select (w/o matching map) learns to pay attention to different features in the satellite image for different vertical cells in the street image. This is consistent with the hypothesis that we can learn to select objects in the satellite image that correspond to a specific vertical cell row and make discriminative cell descriptors that can be used for localizing the street camera in the satellite image. In addition, the attention maps of SliceNet Select (matching map) show that we can learn to pay attention to certain features for a specific street vertical cell in the satellite image. This is consistent with the hypothesis that we can learn to select objects in the satellite image that correspond to a specific vertical cell of a street slice and make discriminative cell descriptors that can be used for localizing the street image in the satellite image.

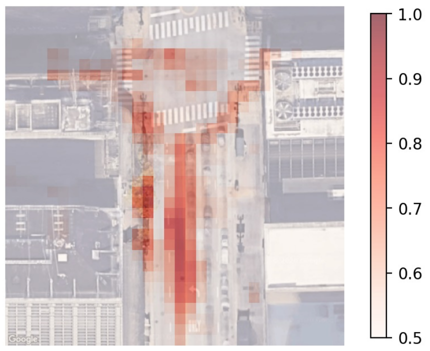
(a) Street image (image pair 1)



(b) Satellite image (image pair 1)

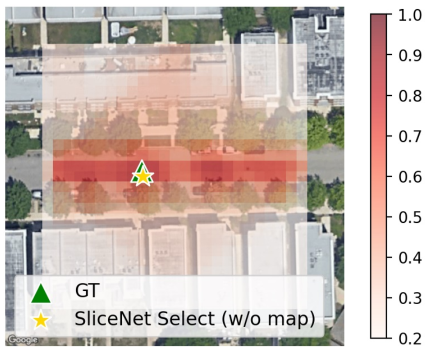


(c) Vertical cell 1 (image pair 1)

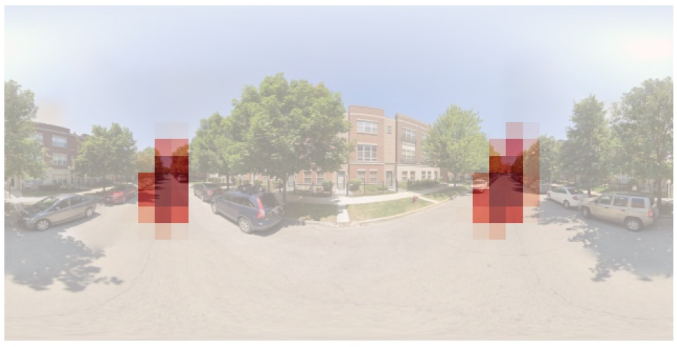


(d) Vertical Cell 1 (image pair 1)

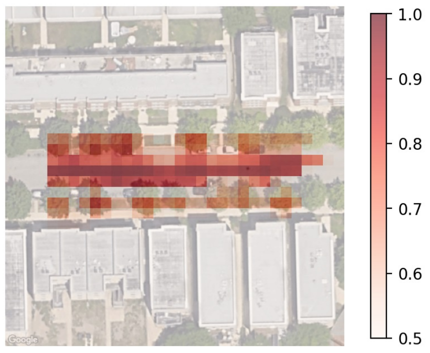
(e) Street image (image pair 2)



(f) Satellite image (image pair 2)

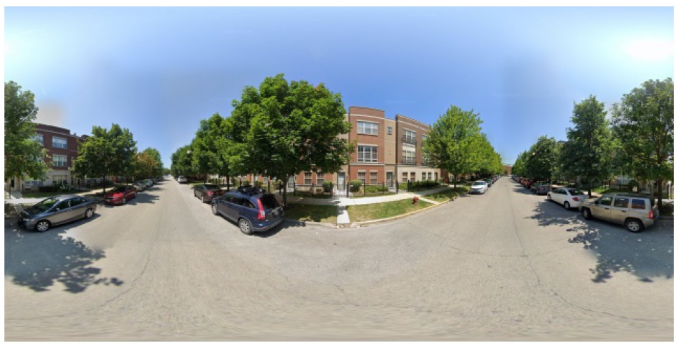


(g) Vertical Cell 1 (image pair 2)

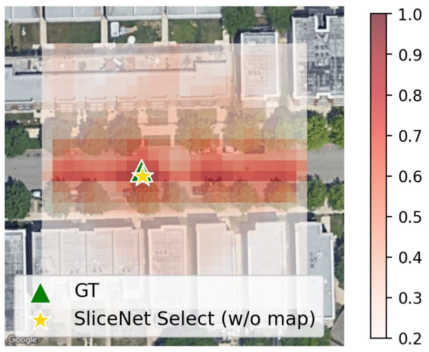


(h) Vertical Cell 1 (image pair 2)

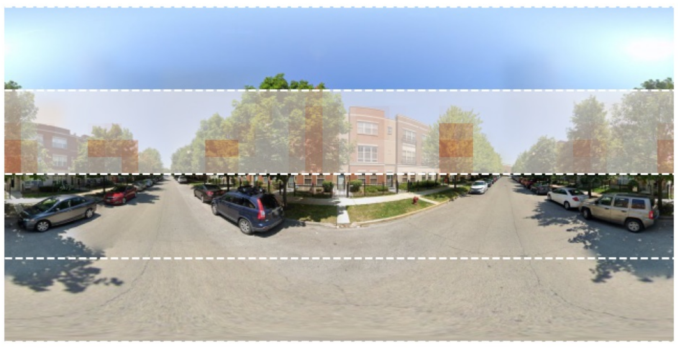
Figure 4.7: Image pair 1 and 2 with the corresponding similarity map in overlay on the satellite image and the attention maps. The model that has been used is SliceNet Select (w/o matching map) with eight horizontal slices and one vertical cell. All street attention maps use the same color bar as the satellite attention maps. All image pairs are aligned. The images are part of the test split of Chicago (same-area) of the VIGOR dataset [107].



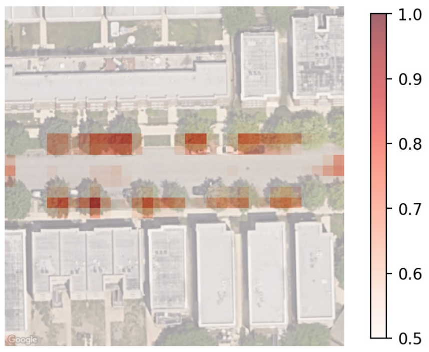
(a) Street image



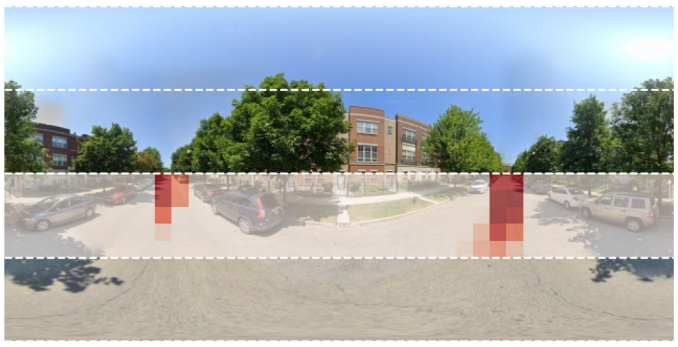
(b) Satellite image



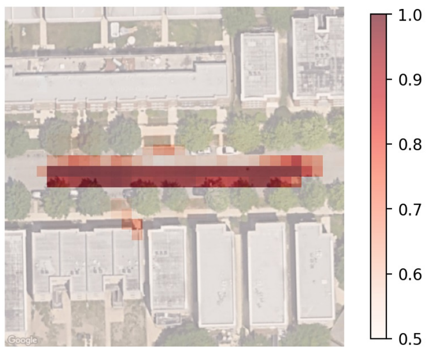
(c) Vertical Cell 2



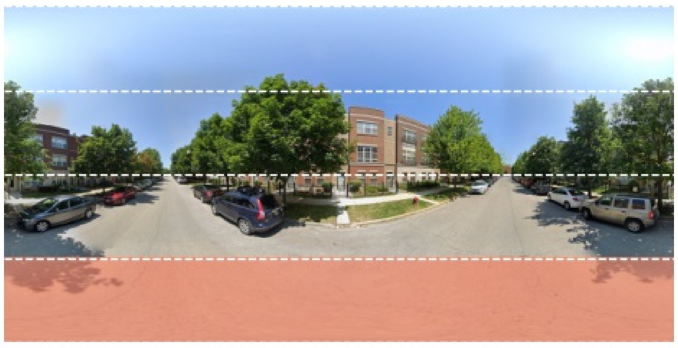
(d) Vertical Cell 2



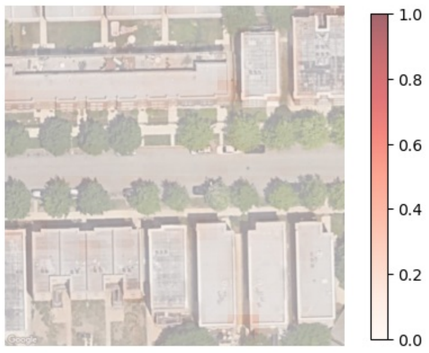
(e) Vertical Cell 3



(f) Vertical Cell 3

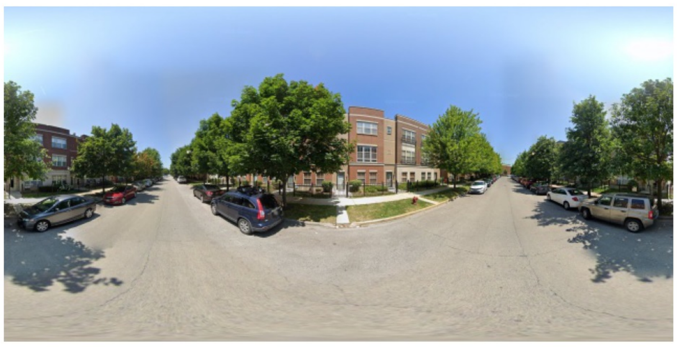


(g) Vertical Cell 4

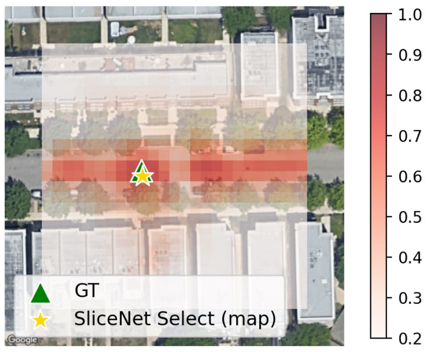


(h) Vertical Cell 4

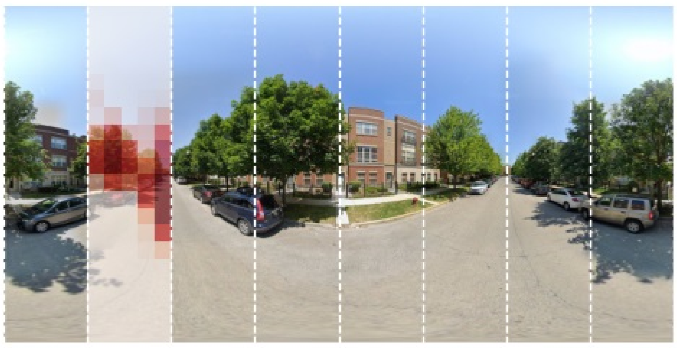
Figure 4.8: Image pair 2 with the corresponding similarity map in overlay on the satellite image and the attention maps. The model that has been used is SliceNet Select (w/o matching map) with eight horizontal slices and four vertical cells. All street attention maps use the same color bar as the satellite attention maps. The image pair is aligned. The images are part of the test split of Chicago (same-area) of the VIGOR dataset [107].



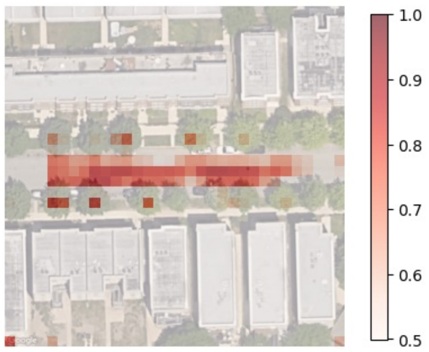
(a) Street image



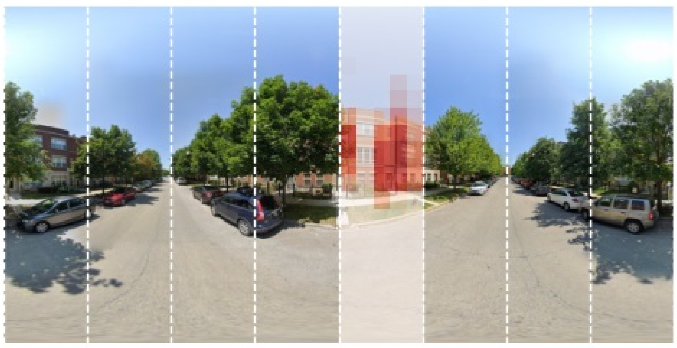
(b) Satellite image



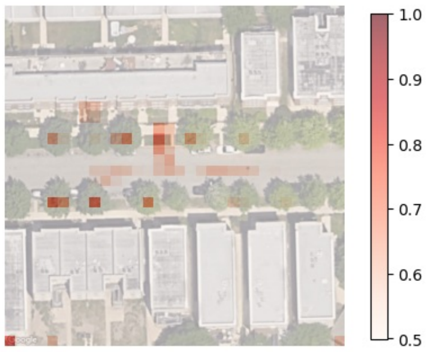
(c) Vertical Cell 2



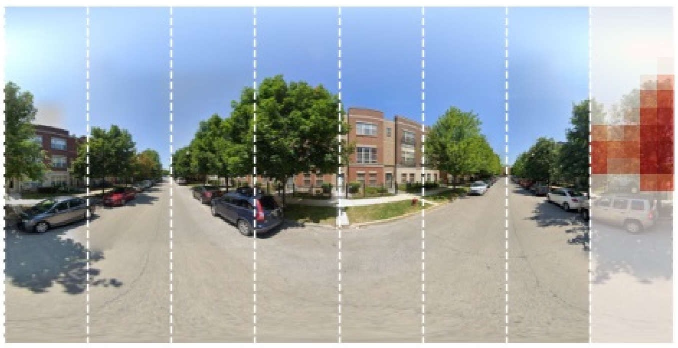
(d) Vertical Cell 2



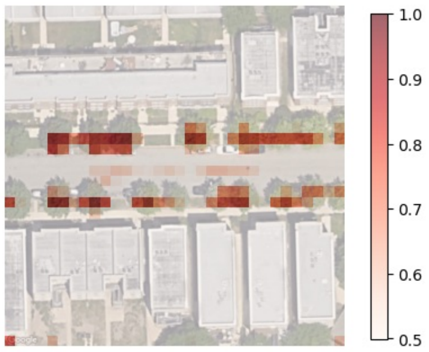
(e) Vertical Cell 3



(f) Vertical Cell 3

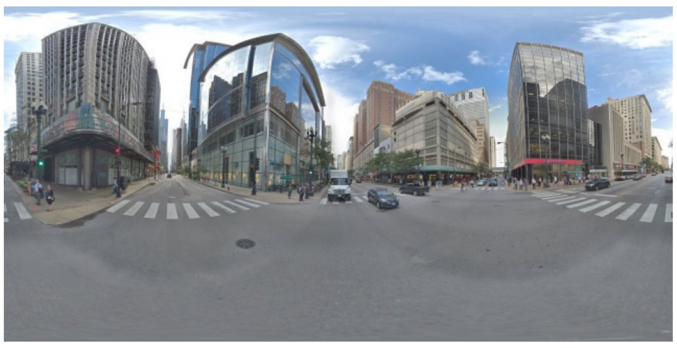


(g) Vertical Cell 4

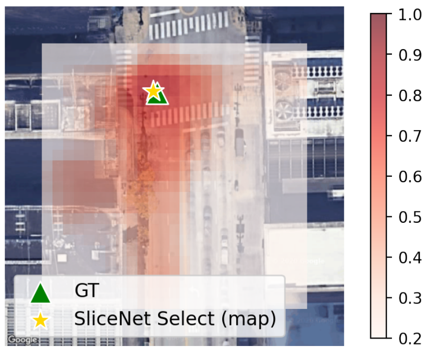


(h) Vertical Cell 4

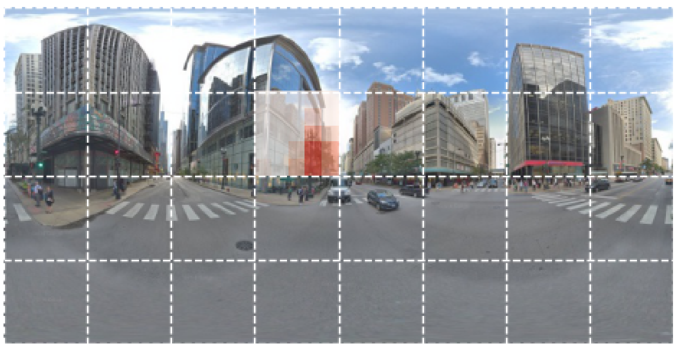
Figure 4.9: Image pair 2 with the corresponding similarity map in overlay on the satellite image and the attention maps. The model that has been used is SliceNet Select (matching map) with eight horizontal slices and one vertical cell. All street attention maps use the same color bar as the satellite attention maps. The image pair is aligned. The images are part of the test split of Chicago (same-area) of the VIGOR dataset [107].



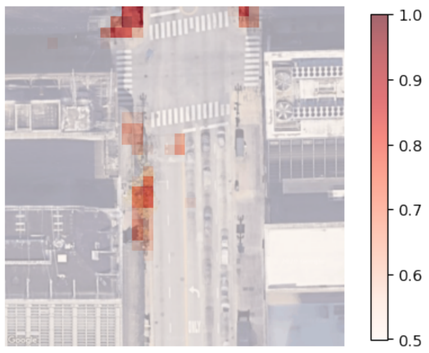
(a) Street image



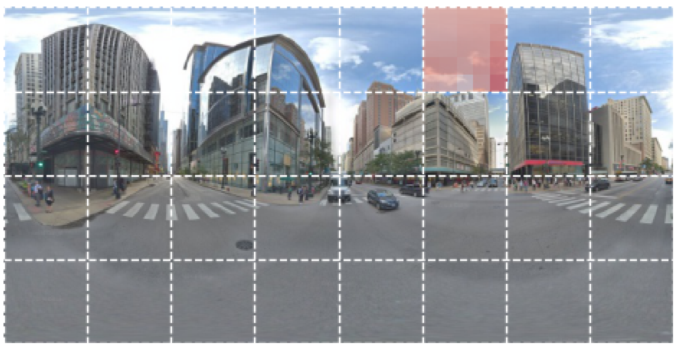
(b) Satellite image



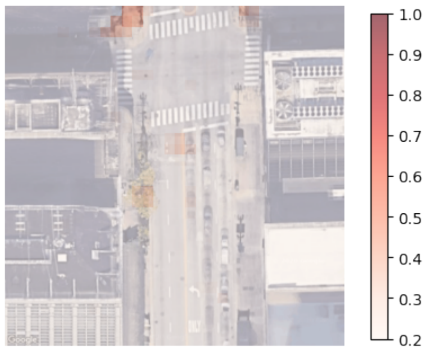
(c) Vertical Cell 2



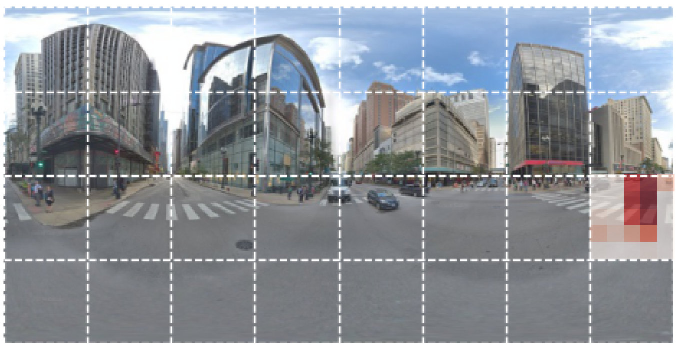
(d) Vertical Cell 2



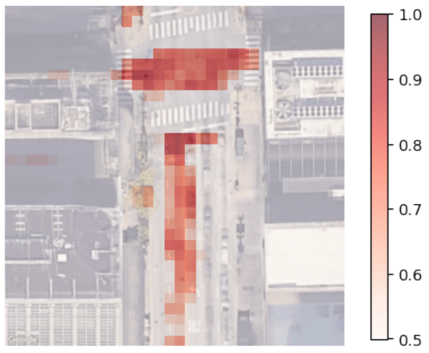
(e) Vertical Cell 3



(f) Vertical Cell 3



(g) Vertical Cell 4



(h) Vertical Cell 4

Figure 4.10: Image pair 1 with the corresponding similarity map in overlay on the satellite image and the attention maps. The model that has been used is SliceNet Select (matching map) with eight horizontal slices and four vertical cells. All street attention maps use the same color bar as the satellite attention maps. The image pair is aligned. The images are part of the test split of Chicago (same-area) of the VIGOR dataset [107].

4.6. Generalization in the Same Area and Across Areas

This section compares SliceNet with the two baselines (see Section 3.2): the CVR model [107] and the MCC model [92]. Below, the experiment is first explained. This is followed by the quantitative and qualitative results. This section ends with preliminary conclusions.

Experiment

The same-area and cross-area settings of the VIGOR dataset are used to compare SliceNet to the baselines. As described in Subsection 4.1.1, we corrected the ground truth labels of the VIGOR dataset because the original labels contained an error. This means that the baselines also need to be retrained. For training the baselines, a PyTorch implementation for the CVR model was created, and the original TensorFlow implementation of the MCC model was used. The CVR model and the MCC model were trained according to the original implementation details.

Several versions were trained for SliceNet. First, SliceNet Basic with eight horizontal slices was trained since this model was used for the proof of concept of SliceNet and was used as a reference for the performance of SliceNet Select. In addition, two versions of SliceNet Select (matching map) were trained: (1) a version with eight horizontal slices and one vertical cell and (2) a version with twenty horizontal slices and one vertical cell. It follows from the ablation study with the number of horizontal slices that the localization performance increases if more horizontal slices are used. However, this increase stagnates for about twenty horizontal slices. Therefore, it has been decided to use twenty horizontal slices for the best model. It follows from the ablation study on the number of vertical cells and the matching map that the best performance is achieved with one vertical cell and that the matching map for each configuration of SliceNet Select tested improves localization performance. Therefore, the matching map is used for both versions of SliceNet Select.

Quantitative results

Tab. 4.6 shows the results for the same-area and cross-area settings. It can be seen that SliceNet Select with twenty horizontal slices achieves the best localization performance for the same-area setting and the best median localization error for the cross-area setting. The mean localization error of SliceNet Select with twenty horizontal slices is slightly higher than that of the CVR model.

Xia et al. [92] used the original ground truth labels to compare their MCC model with the coarse localization and the CVR model. They obtained with their MCC model for the original labels the best mean and median localization error for the same-area setting and the best median localization error for the cross-area setting. The CVR model had a slightly better mean localization error for the cross-area setting. This same trend can be seen in Tab. 4.6. The MCC model again has a better mean and median localization error for the same-area setting and a better median localization error for the cross-area setting. Also, the CVR model has a better mean localization error for the cross-area setting this time. Therefore, it can be concluded that the performances for the retrained CVR model and MCC model are consistent with the performances achieved for the original ground truth labels.

All trained versions of SliceNet achieve better performance than the coarse localization for both the same-area and cross-area settings. For SliceNet, SliceNet Select with eight horizontal slices achieves

Table 4.6: Localization error for same-area and across area settings on the VIGOR dataset [107]. Best performance in **bold**. *Coarse localization* denotes using the satellite view image center as the prediction.

	Same-area		Cross-area	
	Localization error (m)		Localization error (m)	
	Mean	Median	Mean	Median
Coarse localization	14.23	14.89	14.15	14.77
CVR model [107]	7.08	5.70	**7.91**	6.65
MCC model [92]	6.94	3.64	9.05	5.14
SliceNet Basic ($N = 8$)	7.75	4.51	9.08	5.84
SliceNet Select ($N = 8$; $M = 1$)	6.33	3.98	8.50	5.44
SliceNet Select ($N = 20$; $M = 1$)	**5.85**	**3.13**	7.99	**4.93**

lower errors than SliceNet Basic for both settings, and SliceNet Select with twenty horizontal slices performs even better. SliceNet Basic performs similarly to the CVR model but performs less well than the MCC model. In contrast, SliceNet Select with twenty horizontal slices achieves a mean and median localization error for the same-area setting of more than one meter and half a meter less than the MCC model. Also, for the cross-area setting, SliceNet Select significantly outperforms the MCC model. However, it achieves a lower mean localization error than the CVR model.

Xia et al. noted that the CVR model tends to output the average between the visually similar areas, which can result in a location near the center, but that is intuitively unreasonable, e.g., within some vegetation, even though it may have a smaller distance to the ground truth location. This may be the reason that the CVR model achieves a lower mean localization error for the cross-area setting than SliceNet Select with twenty horizontal slices. This explanation is consistent with the significantly lower median localization error obtained by SliceNet. It follows from the localization error distribution that SliceNet has a low localization error for a relatively large number of the image pairs, but it sometimes predicts a location far from the ground truth location.

Qualitative results: success cases

Fig. 4.11 shows four image pairs with the corresponding similarity map in overlay on the satellite image. SliceNet Select with twenty horizontal slices and one vertical cell has been used for the predictions. The predictions of the CVR model and the MCC model are also plotted. However, the probability distribution of the MCC model is not shown.

The first two image pairs of Fig. 4.11 belong to the same-area setting, while the other two image pairs belong to the cross-area setting. It can be seen in Fig. 4.11b that both the CVR model and the MCC model predict a location on the roof of the building, while SliceNet predicts a location near the ground truth location. In addition, SliceNet predicts a multi-modal output, and the second peak is a visually similar place.

In Fig. 4.11d, it can be seen that SliceNet has relatively high similarity along the road and predicts a location near the ground truth location. In contrast, the MCC model predicts a location at the location of a tree.

Fig. 4.11f shows a satellite image where SliceNet and the MCC model both predict near the ground truth, while the CVR model chooses a location near the center of the satellite image. Remarkably, from the location predicted by the CVR model, the trees on the street image can also be seen, however, from a different direction. In SliceNet, the orientation information is used to create the image descriptors so that a specific part of the descriptor indicates that trees should be visible in the western direction. In the descriptors of the CVR model, this orientation information is not explicitly encoded.

Finally, Fig. 4.11h shows that SliceNet predicts close to the ground truth location, while both the CVR model and the MCC model predict a location that is not visually similar to the ground truth location.

Qualitative results: failure cases

Fig. 4.12 shows some failure cases of SliceNet. Again, the first two image pairs are part of the same-area setting, and the last two image pairs are part of the cross-area setting. SliceNet predicts for the first image pair (see Fig. 4.12a and Fig. 4.12b) a location that is far from the ground truth location, while both the CVR model and the MCC model predict a location close to the ground truth location. This image pair is challenging because of the trees. However, road turnings can be seen on the street image, while they cannot be seen at the location that SliceNet predicts. This indicates that SliceNet may be missing a global understanding of the street image.

The second image pair shows a location at an intersection, however SliceNet predicts a location far ahead of the intersection (see Fig. 4.12c and Fig. 4.12d). The third street image (see Fig. 4.12e) shows a traffic circle close to the street camera, but SliceNet predicts a location far from the traffic circle. However, SliceNet does have a second peak in the similarity map near the traffic circle.

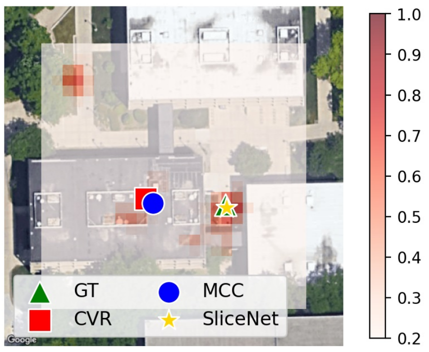
For the last image pair, SliceNet also predicts a location that is far from the ground truth location (see Fig. 4.12g and Fig. 4.12h). Moreover, a lot of uncertainty is visible in the similarity map. Although the predicted location is visually similar to the ground truth location, the right part of the street image could have been used to know that the predicted location is incorrect. In the street image, buildings can be seen through the trees, while there are no buildings behind the trees at the predicted location by SliceNet (East direction). More visualization can be found in Fig. B.1, Fig. B.2, and Fig. B.3.

Preliminary conclusions

It can be concluded that SliceNet achieves a lower mean and median localization error for the same-area setting than the baselines. In addition, a lower median localization error is achieved for the cross-area setting. However, the mean localization error of the CVR model is slightly lower than that of SliceNet. This can be explained by the fact that SliceNet sometimes predicts a multi-model output, while the CVR model may predict the center between two visually similar places [92]. As a result, the CVR model could have a lower mean error but a higher median error. This is consistent with the results from Tab. 4.6. In addition, it can be seen on the visualizations that the CVR model indeed sometimes predicts a location between two similarity peaks from SliceNet.

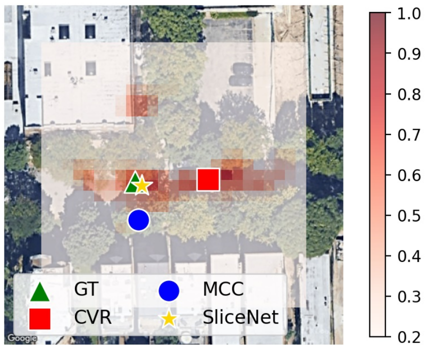
SliceNet's failure cases indicate that SliceNet may lack a global understanding of the local surroundings. For some image pairs, information may be seen in the street image that contradicts the location predicted by SliceNet. SliceNet descriptors consist of slice descriptors, each containing the encoding of a single slice. However, the SliceNet architecture does not include an operation for modeling global connections as can be done with self-attention [81], for example.

(a) Street image (image pair 4)



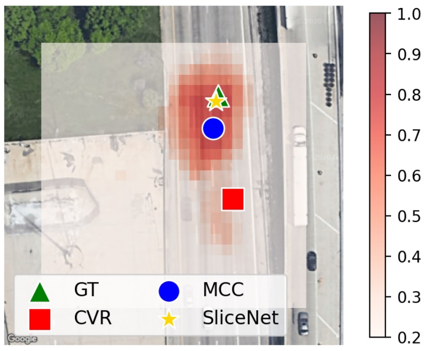
(b) Satellite image (image pair 4)

(c) Street image (image pair 5)

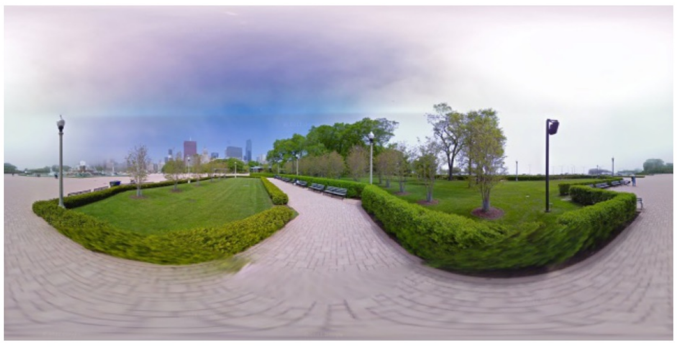


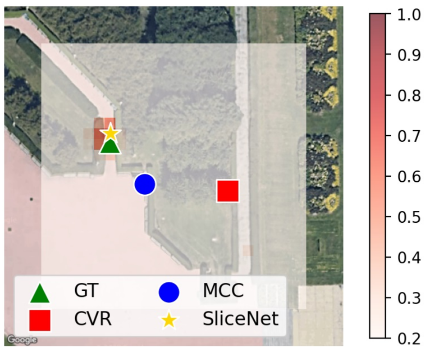
(d) Satellite image (image pair 5)

(e) Street image (image pair 6)



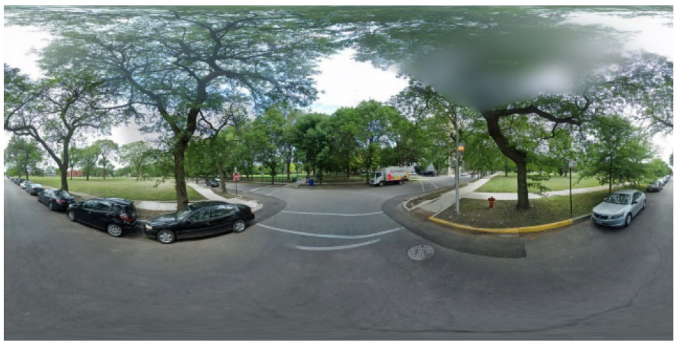
(f) Satellite image (image pair 6)


(g) Street image (image pair 7)

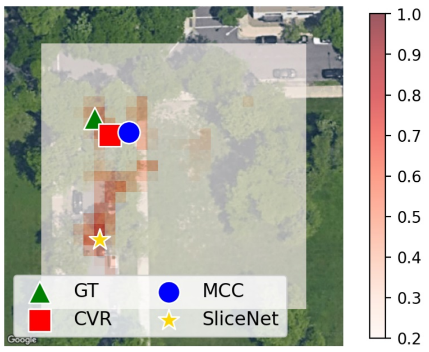


(h) Satellite image (image pair 7)

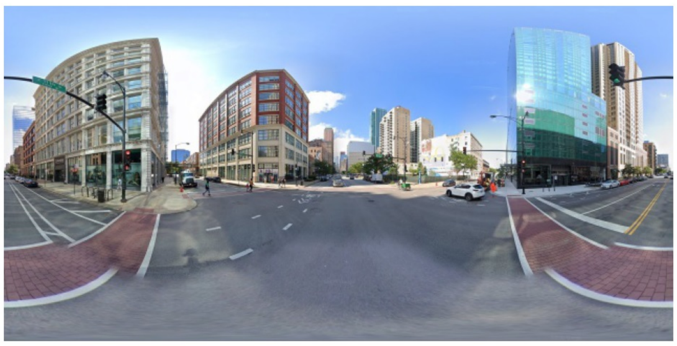
Figure 4.11: Image pairs 4, 5, 6, and 7 with the corresponding similarity map in overlay on the satellite image. The model that has been used is SliceNet Select (matching map) with twenty horizontal slices and one vertical cell. The first two image pairs are part of the same-area setting, and the last two image pairs are part of the cross-area setting. All image pairs are aligned. The images are part of the VIGOR dataset [107].



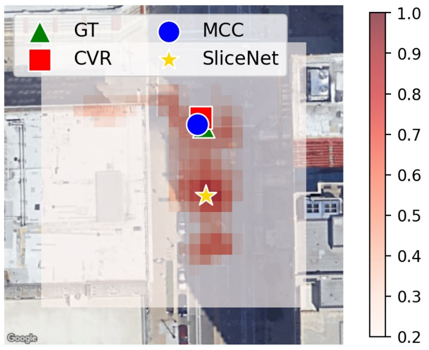
(a) Street image (image pair 8)



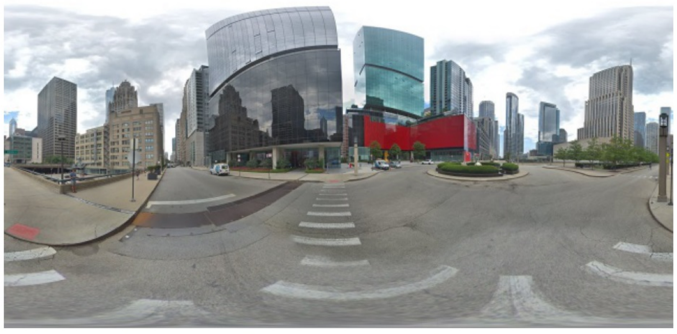
(b) Satellite (image pair 8)



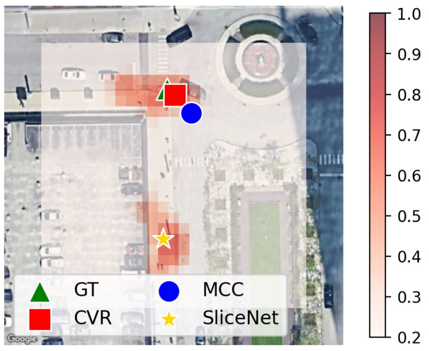
(c) Street image (image pair 9)



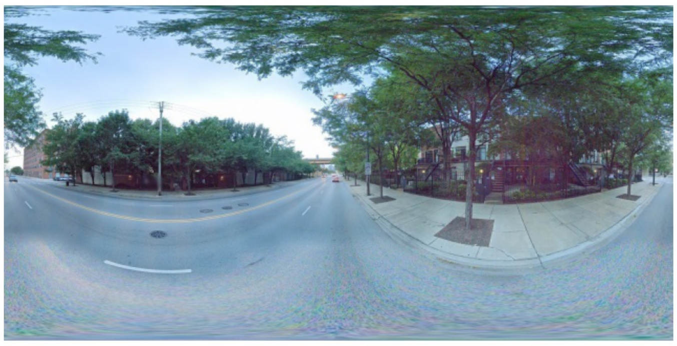
(d) Satellite image (image pair 9)



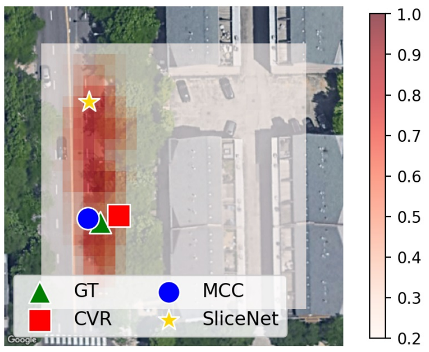
(e) Street image (image pair 10)



(f) Satellite image (image pair 10)



(g) Street image (image pair 11)



(h) Satellite image (image pair 11)

Figure 4.12: Image pairs 8, 9, 10, and 11 with the corresponding similarity map in overlay on the satellite image. The model that has been used is SliceNet Select (matching map) with twenty horizontal slices and one vertical cell. The first two image pairs are part of the same-area setting, and the last two image pairs are part of the cross-area setting. All image pairs are aligned. The images are part of the VIGOR dataset [107].

4.7. Runtime Performance en Number of Parameters

This section compares the runtime performance and number of parameters of SliceNet with those of the baselines. In Section 4.6, it has been shown that SliceNet Select with twenty horizontal cells achieves better median localization performance than the baselines for the same-area and the cross-area settings of the VIGOR dataset [107]. However, runtime performance is also important as stated in Subsection 3.2.1.

In order to compare runtime performance, two aspects are considered: (1) the inference speed for a single image pair and (2) the GPU memory usage. An update frequency of at least 100 Hertz is required for the autonomous vehicle [58], and thus the inference speed of the network for a single image pair should be as low as possible. In addition, it is advantageous if the GPU memory usage is low because then more memory can be used for other algorithms of the autonomous vehicle. The number of learnable parameters of each network is also determined. This gives an indication of the complexity of the network.

The versions of SliceNet and baselines used to test the generalization for the same-area and cross-area settings were all tested on an NVIDIA V100 GPU with 32GB of memory. PyTorch 1.12.0 was used to implement SliceNet and the CVR model, while TensorFlow 1.14 was used to implement the MCC model. In order to test the inference speed, the test dataset from the same-area setting was used with a batch size equal to one. A forward pass was then done 1000 times, and the time of each forward was measured. The experiment results, along with the GPU memory usage and the number of learnable parameters, are summarized in Tab. 4.7.

SliceNet Basic achieves the lowest mean and median inference speed of all models tested. In addition, each version of SliceNet is faster than the baselines. The mean inference speed of SliceNet Basic corresponds to about 230 frames per second. Thus, this is considerably more than the required 100 frames per second. SliceNet Select with twenty horizontal slices also achieves more than 100 frames per second, at about 170 frames per second. The CVR model uses by far the least GPU memory. In contrast, all versions of SliceNet have a significantly lower number of learnable parameters than the baselines. Interestingly, the MCC model performs worse for the inference speed, the GPU memory usage, and the number of learnable parameters than the CVR model and SliceNet. However, the MCC model is the only model implemented in TensorFlow, making comparison difficult. Besides, the inference speed and GPU memory usage depend on the implementation of the models.

Table 4.7: The runtime performance and the number of learnable parameters of SliceNet and baselines. Best performance in **bold**. *GPU memory* denotes the GPU memory usage, and *Parameters* denotes the number of learnable parameters of the network.

Method	Inference speed (ms)		GPU memory (MB)	Parameters
	Mean	Median		
CVR model [107]	6.4	6.3	**690**	47,155,714
MCC model [92]	35.6	35.7	8,161	79,696,881
SliceNet Basic ($N = 8$)	**4.3**	**4.3**	1,113	**29,429,376**
SliceNet Select ($N = 8$; $M = 1$)	5.3	5.3	1,118	29,497,284
SliceNet Select ($N = 20$; $M = 1$)	5.9	5.8	1,118	29,497,284

5 Conclusion

The introduction stated that about 1.3 million people die each year due to road traffic crashes and that road traffic injuries are even the leading cause of death for children and young adults aged 5-29 years. Autonomous vehicles could play a significant role in assuring road safety, as more than 90 percent of all accidents are caused by human error. Besides, automated driving could improve traffic flows, promote the mobility and productivity of people, and improve driving comfort.

High-definition maps are crucial for safe automated driving as they contain a detailed description of the vehicle's surroundings. In order to use these maps, an accurate localization technique is required to provide vehicle positioning in map coordinates. GNSS-based localization techniques can provide global localization. However, the localization performance is poor in urban regions due to the presence of tall buildings. As a result, there can be localization errors in the order of tens of meters. These errors are in strong contrast with the requirements for automated driving systems based on the road geometry standards of the United States. The required lateral en longitudinal error bounds for vehicles operating on freeway roads are 0.57 and 1.40 meters, respectively. Besides, the road geometry makes requirements even more stringent on local streets with lateral and longitudinal error bounds of 0.29 meters.

However, onboard vehicle sensors can be used to obtain a more accurate localization. This process consists of two steps. The first step is to get a coarse localization for the vehicle using any rough localization prior, and the second step is to use visual localization to improve the vehicle's localization and get a fine localization. In this work, cross-view image matching-based localization has been considered for obtaining fine localization. Recent cross-view matching work proposed to localize the street camera in a single satellite image to get an accurate localization, but these methods achieved localization errors in the order of several meters. Although this can improve the coarse localization, the performance is not yet high enough for using it to localize autonomous vehicles.

We observed that existing cross-view image metric localization methods create a global street image descriptor and one or multiple satellite descriptors, and use these for predicting the camera location. In contrast, structure-based methods employ local feature matching to estimate the 6-DoF street camera pose and achieve centimeter-level accurate localization. Hence, we wondered whether something similar could be done for cross-view matching, as this may be the key to improving the localization performance of cross-view matching methods.

Unfortunately, the viewpoint difference for cross-view image matching is considerably more significant than for structure-based matching, making matching local descriptors between the street and the satellite view more difficult. For structure-based methods, one-to-one correspondences between a query and the reference image are available during training. In contrast, explicitly labeled correspondences between the two views are unavailable for cross-view matching while learning local descriptors requires correspondence between the two views.

However, we noted that the geometric relationship between the street and satellite view can be used to find local correspondences. The relationship implies that every vertical line in the street image has a corresponding azimuth direction in the satellite image. Line pairs can be used to delineate an image slice in both images, and two belonging slices have a visual overlap.

We hypothesized that the image slices can be used for learning local correspondences between the views. However, we noted that it is an open problem how this visual overlap can be used to learn the correspondences. In addition, we stated that some objects cannot be used for matching, and therefore, there is a need to select sparse features from the images. Besides, we noted that it is an open problem how the local image features can be matched and how the matching can be used to determine the street camera location. Based on the identified open problems, the main research question of this work, as stated in the introduction (see Chapter 1), was formulated as:

How can we end-to-end learn sparse local image features across the street and the satellite view without using explicitly labeled correspondences, and how can this be used for localizing the street camera in the satellite image?

In order to answer the main research question, three subquestions were researched:

1. How can we end-to-end learn local image features across the street and satellite view without using explicitly labeled correspondences?
2. How can we extract sparse features from the street and satellite view?
3. How can we use local image feature matching for localizing the street camera in the satellite image?

In Chapter 2, the literature related to cross-view matching, local image feature matching, and exploiting the azimuth prior has been reviewed. We noted that existing cross-view image metric localization methods do not learn explicit local correspondences between the street and satellite view and do not use the azimuth prior for improving the localization performance. Besides, we concluded that local image feature matching methods use one-to-one correspondences during training, while these are unavailable for cross-view matching. Lastly, we observed that the azimuth prior has been used to improve the performance of cross-view image retrieval methods and related fields for mapping images to another view. However, the prior has not been used for learning local correspondences between the two views.

Subsequently, in Chapter 3, the methodology used to answer the research questions has been explained. In that chapter, we first stated the problem definition of street-to-satellite image metric localization. After that, we explained how sparse local matches between the street and satellite views could be used to determine the street camera location. Then, we described the baselines against which we compared our proposed method and explained our plan for learning local correspondences between the views. Finally, we presented the architecture of SliceNet and compared it with that of the baselines.

Lastly, the experimental results have been presented in Chapter 4. First, it was motivated why the VIGOR dataset [107] is used to evaluate the methods, and it was stated what metrics are used to measure the methods' performances. After that, the implementation details of SliceNet were described, and the proof of concept experiment for SliceNet was presented. Then, an ablation study on the effect of the number of horizontal and vertical image slices on localization performance was presented. Subsequently, SliceNet was compared to baselines for generalization in the same-area and across-areas. Finally, the runtime performance and the number of parameters of SliceNet and the baselines were compared.

The answers to the three subquestions are described below. This is followed by the answer to the main research question.

First subquestion: *How can we end-to-end learn local image features across the street and satellite view without using explicitly labeled correspondences?*

The horizontal field of view of the street camera can be divided into N equally-sized slices using the azimuth prior (see Subsection 3.3.1). This prior states that each vertical line in the street image has a corresponding azimuth direction in the satellite image. This allows the street image to be divided into N rectangular image slices using vertical lines. In combination with the street camera pose, the azimuth directions of these vertical lines can be used to divide the satellite image into N pie-like image slices. Matching slices have a visual overlap. Slice descriptors can be created for each slice, and a similarity measure can be used to measure the similarity between two descriptors.

We hypothesized that visual overlap can be used to learn that matching slice descriptors are similar while non-matching slice descriptors are dissimilar (see Subsection 3.3.2). This was called the orientation hypothesis. We noted that creating the satellite slices requires the street camera pose. However, other locations in the satellite image can also be used to create the satellite slices. Therefore, we hypothesized that the visual overlap between matching slices can be used to learn that the satellite slice descriptors of the camera location are more similar to the street slice descriptors than the satellite slice descriptors of any other location in the satellite image (see Subsection 3.3.3). This was called the location hypothesis.

The orientation hypothesis relies on the assumption that the camera pose is available for slicing in the satellite image, while the location hypothesis assumes that the orientation is known for slicing. The pose hypothesis was proposed as the next step, namely that both the location and orientation of the camera are not used for slicing in the satellite image. We hypothesized that the visual overlap can be used to learn that the satellite slice descriptors of the camera pose are more similar to street slice descriptors than the satellite slice descriptors of other poses in the satellite image (see Subsection 3.3.4).

In order to test these hypotheses, SliceNet Basic was developed (see Section 3.4). SliceNet Basic extracts deep features from both images, uses horizontal slicing to divide the feature maps into slices, and determines a descriptor for each slice. The infoNCE loss [52] was used to learn that matching slice descriptors must be similar and non-matching slice descriptors must be dissimilar. In order to test each hypothesis, different negative samples were used for the loss function.

The results show that SliceNet Basic can learn orientation, location, and pose discriminative local features (see Section 4.4). Therefore, it is concluded that the azimuth prior, combined with horizontal slicing, can be used to learn end-to-end local image features across the street and satellite view without using explicitly labeled correspondences.

Second subquestion: *How can we extract sparse features from the street and satellite view?*

We noted that a street slice can be divided into vertical cells based on the semantic content visible in the slice and called this vertical slicing. In addition, we stated that this observation could be used for structuring the slice descriptor (see Subsection 3.3.5). A separate descriptor can be created for each cell of a street slice, and the cell descriptors can be combined to create the street slice descriptor. However, satellite slices cannot be partitioned in a similar way. However, matching objects in the satellite image can be selected for each vertical cell of the street image. We hypothesized that we can learn to select objects in the satellite image that belong to a particular vertical cell row (see Subsection 3.3.5). In addition, we argued that matching features can be searched for a single vertical cell rather than a vertical row. Therefore, we hypothesized that we can learn to select objects in the satellite image that belong to a particular vertical cell (see Subsection 3.3.6).

In order to test these hypotheses, SliceNet Select was developed (see Section 3.4). This is an extension of SliceNet Basic, where each branch has an attention module to reweigh the features of a feature map. This allows the network to give important features a higher weight than unimportant features. In addition, SliceNet Select includes a cross-branch connection so that the street cell descriptor can be used to select features in the satellite feature map.

It follows from the results that without the cross branch connection, SliceNet Select learns to pay attention to different features in the satellite image for different vertical cell rows in the street image. This is consistent with the hypothesis that we can learn to select objects in the satellite image that correspond to a specific vertical cell row. In addition, the attention maps of SliceNet Select with the cross branch connection show that we can learn to pay attention to certain features for a specific street vertical cell in the satellite image. This is consistent with the hypothesis that we can learn to select objects in the satellite image that correspond to a specific vertical cell of a street slice. Therefore, it is concluded that attention can be used to extract sparse features for the street and satellite view.

Third subquestion: *How can we use local image feature matching for localizing the street camera in the satellite image?*

For the first subquestion, it is described that horizontal slicing can be used to create image slices, and a slice descriptor can be created for each slice. The slice descriptors of a view can be concatenated using the spatial structure of the slicing. This results in a street image descriptor and a satellite image descriptor. The satellite descriptor depends on the pose used in the satellite image for the slicing. Therefore, the similarity between the street image descriptors and satellite image descriptors can be used for localization. It can be learned that the satellite descriptor made with the street camera pose should be more similar to the street image descriptor than the satellite image descriptors made with other poses. During inference, different poses can be evaluated, and the true camera pose can be estimated by choosing the pose whose satellite image descriptor has the highest similarity.

The effect of the number of horizontal image slices on localization performance was studied by training SliceNet Basic for different values for the number of horizontal slices (see Subsection 4.5.1).

The results show that the localization error decreases when more horizontal slices are used. In addition, the qualitative results show that less uncertainty is present for the localization when more horizontal slices are used. Both the quantitative and the qualitative results are in line with the expectation that the localization performance improves when more horizontal slices are used because when more horizontal slices are used, a small change of the location reduces the overlap between matching slices more. However, the increase in localization performance does plateau when more than about 20 horizontal slices are used.

In addition, the effect of the number of vertical cells on localization performance was studied by training SliceNet Select for different values for the number of vertical cells (see Subsection 4.5.2). The results show that using the attention map improves localization performance, and the best performance is achieved for one vertical cell. The localization performance decreases monotonically when the number of vertical cells increases. Lastly, using the street cell descriptor to create the satellite attention map, i.e., the cross branch connection, improves the localization performance.

Main research question: How can we end-to-end learn sparse local image features across the street and the satellite view without using explicitly labeled correspondences, and how can this be used for localizing the street camera in the satellite image?

The answer to the main research questions is obtained by combining the answers of the subquestions. The azimuth prior can be used to divide the street and satellite image into image slices. Each street slice has a corresponding satellite slice and a pair of slices has a visual overlap. Subsequently, a slice descriptor can be created for each slice, and the slice descriptors of a view can be combined into image descriptors. The visual overlap between matching slices can be used to learn discriminative descriptors. It can be learned that the satellite image descriptor created with the camera pose is more similar to the street image descriptor than a satellite image descriptor created with a different pose. During inference, the similarity can be determined for predetermined poses, and the pose with the highest similarity is used as the prediction. Besides, attention masks can be used to reweigh the feature maps of the images to select features. Therefore, we conclude that we can learn sparse local image features across the street and satellite view end-to-end without using explicitly labeled correspondences, and the camera can be located in the satellite image. Moreover, the orientation of the street image can be predicted if the street and the satellite image are not aligned.

In extensive experiments on the VIGOR dataset [107], we have shown that local image descriptors can be learned to localize the street camera. After that, we showed that the localization performance can be improved by using more horizontal slices. Then, we showed that SliceNet can learn to attend to specific parts of the satellite image based on the content of the street image to build more discriminative slice descriptors.

We achieve state-of-the-art performance for metric localization on the VIGOR dataset. Notably, our method reduces the median metric localization error by 21% and 4% compared to the state-of-the-art methods when generalizing, respectively, in the same area and across areas. In addition, SliceNet achieves a higher inference speed than the state-of-the-art methods, and it has significantly fewer learnable parameters. Finally, SliceNet is the first street-to-satellite image metric localization method that can output a dense distribution for the 3-DoF pose of the street camera.

The state-of-the-art methods create a global image descriptor for the street image and match it with one or more satellite descriptors. This combines all the information from the street image into a single encoding, and the matching is done implicitly. These methods can learn priors about the geometric relationship between the street and satellite image through the training objective. However, there is no guarantee that the matching of these methods satisfies that relationship. In contrast, SliceNet concatenates the slice descriptors, making an image's descriptor clearly structured. This ensures that only slice permutations that satisfy the geometric relationship between the views are matched.

SliceNet's failure cases indicate that SliceNet may lack a global understanding of the local surroundings. For some image pairs, information may be seen in the street image that contradicts the location predicted by SliceNet. The location predicted by SliceNet is the location for which the highest similarity is obtained. This means that if relatively many of the satellite slice descriptors match well with the street slice descriptors, a location may be predicted for which a single street slice descriptor does not match. However, slice descriptors do not contain information about the matching of other slice descriptors.

Despite the fact that SliceNet improves the state-of-the-art localization performance for the VIGOR dataset, the obtained performance does not meet the requirements for localizing autonomous vehicles.

For example, SliceNet achieves a median localization error of more than 3 meters for the same-area setting of the VIGOR dataset, while for local streets, there is an upper bound of 0.29 meters. Therefore, it can be concluded that there is still a large gap between the current performance achieved for street-to-satellite image matching and the performance required for autonomous vehicles.

However, local feature matching is considered the correct direction for the cross-view image metric localization field. The use of local feature matching makes it possible to filter matches based on the geometric relationship between the two views. Moreover, the explicit matches can be used to better understand the matching.

As future work, it is proposed to evaluate SliceNet for the other existing cross-view image metric localization datasets such as the Oxford RobotCar dataset [46, 47, 90], the KITTI dataset [25, 72] and the Ford multi-AV dataset [1, 72]. These datasets contain street images with a limited horizontal field of view. The effect of this on SliceNet's localization performance needs to be investigated.

SliceNet's failure cases suggest that the slice descriptors lack information about the content and matching of the other slice descriptors. Therefore, how this information could be incorporated into the slice descriptor could be investigated. For this purpose, the SuperGlue method [63] could be examined. This method uses self-attention (intra-image) and cross-attention (inter-image) to leverage both the spatial relationships of features and their visual appearance.

Lastly, SliceNet determines the slices for multiple poses in the satellite image and estimates the true camera location by using the location of the pose with the highest similarity. However, it may be more efficient if the satellite image features were matched directly with the street features. For this purpose, inspiration could be used from image matching methods such as SuperPoint [23] and LF-Net [51]. The obtained one-to-one matches between the two views could then be used to determine the street camera location as described in Subsection 3.1.2.

References

[1] Siddharth Agarwal, Ankit Vora, Gaurav Pandey, Wayne Williams, Helen Kourous, and James McBride. “Ford Multi-AV Seasonal Dataset”. In: *IJRR*. 2020, pp. 1367–1376 (cit. on pp. 9, 52).

[2] Mohammad OA Aqel, Mohammad H Marhaban, M Iqbal Saripan, and Napsiah Bt Ismail. “Review of Visual Odometry: Types, Approaches, Challenges, and Applications”. In: *SpringerPlus*. 2016, pp. 1–26 (cit. on p. 1).

[3] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. “NetVLAD: CNN Architecture for Weakly Supervised Place Recognition”. In: *IEEE/CVF CVPR*. 2016, pp. 5297–5307 (cit. on pp. 2, 6).

[4] Mayank Bansal, Harpreet S Sawhney, Hui Cheng, and Kostas Daniilidis. “Geo-Localization of Street Views with Aerial Image Databases”. In: *ACM Multimedia*. 2011, pp. 1125–1128 (cit. on p. 5).

[5] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. “Key.Net: Keypoint Detection by Handcrafted and Learned CNN Filters”. In: *IEEE/CVF ICCV*. 2019, pp. 5836–5844 (cit. on pp. 10, 11).

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. “SURF: Speeded Up Robust Features”. In: *ECCV*. 2006, pp. 404–417 (cit. on p. 5).

[7] Paul R Beaudet. “Rotationally Invariant Image Operators”. In: *IEEE IJCPR*. 1978 (cit. on p. 11).

[8] Boaz Ben-Moshe, Elazar Elkin, Harel Levi, and Ayal Weissman. “Improving Accuracy of GNSS Devices in Urban Canyons”. In: *CCCG*. 2011, pp. 511–515 (cit. on p. 1).

[9] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. “DSAC-Differentiable RANSAC for Camera Localization”. In: *IEEE/CVF CVPR*. 2017, pp. 6684–6692 (cit. on p. 11).

[10] Eric Brachmann and Carsten Rother. “Neural-guided RANSAC: Learning where to Sample Model Hypotheses”. In: *IEEE/CVF ICCV*. 2019, pp. 4322–4331 (cit. on p. 11).

[11] Julia Breßler, Pierre Reisdorf, Marcus Obst, and Gerd Wanielik. “GNSS Positioning in Non-line-of-Sight Context—a Survey”. In: *IEEE ITSC*. IEEE. 2016, pp. 1147–1154 (cit. on p. 1).

[12] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. “Signature Verification using a ”Siamese” Time Delay Neural Network”. In: *NeurIPS*. 1993 (cit. on p. 5).

[13] Matthew Brown, Gang Hua, and Simon Winder. “Discriminative Learning of Local Image Descriptors”. In: *IEEE TPAMI*. 2010, pp. 43–57 (cit. on p. 10).

[14] Sudong Cai, Yulan Guo, Salman Khan, Jiwei Hu, and Gongjian Wen. “Ground-to-Aerial Image Geo-Localization With a Hard Exemplar Reweighting Triplet Loss”. In: *IEEE/CVF ICCV*. 2019, pp. 8391–8400 (cit. on pp. 6, 23).

[15] Bingyi Cao, Andre Araujo, and Jack Sim. “Unifying Deep Local and Global Features for Image Search”. In: *ECCV*. 2020, pp. 726–743 (cit. on pp. 2, 9).

[16] Zetao Chen, Adam Jacobson, Niko Sünderhauf, Ben Upcroft, Lingqiao Liu, Chunhua Shen, Ian Reid, and Michael Milford. “Deep Learning Features at Scale for Visual Place Recognition”. In: *IEEE ICRA*. 2017, pp. 3223–3230 (cit. on p. 2).

[17] Sumit Chopra, Raia Hadsell, and Yann LeCun. “Learning a Similarity Metric Discriminatively, with Application to Face Verification”. In: *IEEE/CVF CVPR*. 2005, pp. 539–546 (cit. on p. 5).

[18] Peter Hviid Christiansen, Mikkel Fly Kragh, Yury Brodskiy, and Henrik Karstoft. “Unsuperpoint: End-to-End Unsupervised Interest Point Detector and Descriptor”. In: *arXiv preprint arXiv: 1907.04011*. 2019 (cit. on pp. 9, 10).

[19] European Commission. *Road*. URL: `https://transport.ec.europa.eu/transport-themes/intelligent-transport-systems/road_en` (visited on 08/31/2022) (cit. on p. 1).

[20] Jin Cui, Lin Shen Liew, Giedre Sabaliauskaite, and Fengjun Zhou. “A Review on Safety Failures, Security Attacks, and Available Countermeasures for Autonomous Vehicles”. In: *Ad Hoc Networks*. 2019, p. 101823 (cit. on p. 1).

[21] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. “Image Retrieval: Ideas, Influences, and Trends of the New Age”. In: *ACM CSUR*. 2008, pp. 1–60 (cit. on p. 6).

[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. “Imagenet: A Large-Scale Hierarchical Image Database”. In: *IEEE/CVF CVPR*. 2009, pp. 248–255 (cit. on p. 29).

[23] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. “Superpoint: Self-Supervised Interest Point Detection and Description”. In: *IEEE/CVF CVPR workshops*. 2018, pp. 224–236 (cit. on pp. 9, 10, 52).

[24] Martin A Fischler and Robert C Bolles. “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”. In: *CACM*. 1981, pp. 381–395 (cit. on p. 11).

[25] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. “Vision meets Robotics: The KITTI Dataset”. In: *IJRR*. 2013, pp. 1231–1237 (cit. on pp. 9, 52).

[26] Chris Harris, Mike Stephens, et al. “A Combined Corner and Edge Detector”. In: *AVC*. 1988, pp. 10–5244 (cit. on p. 11).

[27] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. “Mask R-CNN”. In: *IEEE/CVF ICCV*. 2017, pp. 2961–2969 (cit. on p. 10).

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *IEEE/CVF CVPR*. 2016, pp. 770–778 (cit. on p. 11).

[29] Sixing Hu, Mengdan Feng, Rang MH Nguyen, and Gim Hee Lee. “CVM-Net: Cross-View Matching Network for Image-based Ground-to-Aerial Geo-Localization”. In: *IEEE/CVF CVPR*. 2018, pp. 7258–7267 (cit. on pp. 2, 3, 5, 6, 12).

[30] Joel Janai, Fatma Güney, Aseem Behl, Andreas Geiger, et al. “Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art”. In: *FTCGV*. 2020, pp. 1–308 (cit. on p. 1).

[31] Mahmut Kaya and Hasan Şakir Bilge. “Deep Metric Learning: A Survey”. In: *Symmetry*. 2019, p. 1066 (cit. on p. 6).

[32] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. “Supervised Contrastive Learning”. In: *NeurIPS*. 2020, pp. 18661–18673 (cit. on p. 25).

[33] Diederik P Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *arXiv preprint arXiv:1412.6980*. 2014 (cit. on p. 30).

[34] Florian Kluger, Eric Brachmann, Hanno Ackermann, Carsten Rother, Michael Ying Yang, and Bodo Rosenhahn. “CONSAC: Robust Multi-model Fitting by Conditional Sample Consensus”. In: *IEEE/CVF CVPR*. 2020, pp. 4634–4643 (cit. on p. 11).

[35] Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. “PifPaf: Composite Fields for Human Pose Estimation”. In: *IEEE/CVF CVPR*. 2019, pp. 11977–11986 (cit. on p. 10).

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet Classification with Deep Convolutional Neural Networks”. In: *NeurIPS*. 2012, pp. 1097–1105 (cit. on p. 5).

[37] Songlian Li, Zhigang Tu, Yujin Chen, and Tan Yu. “Multi-Scale Attention Encoder for Street-to-Aerial Image Geo-Localization”. In: *CAAI TIT*. 2022 (cit. on pp. 3, 6, 12).

[38] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. “BEVFormer: Learning Bird’s-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers”. In: *ECCV*. 2022 (cit. on p. 2).

[39] Jinliang Lin, Zhedong Zheng, Zhun Zhong, Zhiming Luo, Shaozi Li, Yi Yang, and Nicu Sebe. “Joint Representation Learning and Keypoint Detection for Cross-view Geo-Localization”. In: *IEEE TIP*. 2022 (cit. on p. 12).

[40] Tsung-Yi Lin, Serge Belongie, and James Hays. “Cross-View Image Geolocalization”. In: *IEEE/CVF CVPR*. 2013, pp. 891–898 (cit. on p. 5).

[41] Tsung-Yi Lin, Yin Cui, Serge Belongie, and James Hays. “Learning Deep Representations for Ground-to-Aerial Geolocalization”. In: *IEEE/CVF CVPR*. 2015, pp. 5007–5015 (cit. on pp. 5, 6).

[42] Liu Liu and Hongdong Li. “Lending Orientation to Neural Networks for Cross-view Geo-Localization”. In: *IEEE/CVF CVPR*. 2019, pp. 5624–5633 (cit. on pp. 3, 6, 7, 12, 13, 16).

[43] David G Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *IJCV*. 2004, pp. 91–110 (cit. on pp. 5, 10).

[44] Jiayi Ma, Xingyu Jiang, Aoxiang Fan, Junjun Jiang, and Junchi Yan. “Image Matching from Handcrafted to Deep Features: A Survey”. In: *IJCV*. 2021, pp. 23–79 (cit. on pp. 9–11, 24).

[45] Jiayi Ma, Xingyu Jiang, Junjun Jiang, Ji Zhao, and Xiaojie Guo. “LMR: Learning a Two-class Classifier for Mismatch Removal”. In: *IEEE TIP*. 2019, pp. 4045–4059 (cit. on p. 11).

[46] Will Maddern, Geoffrey Pascoe, Matthew Gadd, Dan Barnes, Brian Yeomans, and Paul Newman. “Real-time Kinematic Ground Truth for the Oxford RobotCar Dataset”. In: *arXiv preprint arXiv:2002.10152*. 2020 (cit. on pp. 9, 52).

[47] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. “1 year, 1000 km: The Oxford RobotCar dataset”. In: *IJRR*. 2017, pp. 3–15 (cit. on pp. 9, 52).

[48] Krystian Mikolajczyk and Cordelia Schmid. “A Performance Evaluation of Local Descriptors”. In: *IEEE TPAMI*. 2005, pp. 1615–1630 (cit. on p. 10).

[49] Masafumi Noda, Tomokazu Takahashi, Daisuke Deguchi, Ichiro Ide, Hiroshi Murase, Yoshiko Kojima, and Takashi Naito. “Vehicle Ego-localization by Matching In-vehicle Camera Images to an Aerial Image”. In: *ACCV*. 2010, pp. 163–173 (cit. on pp. 5, 12).

[50] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. “Large-Scale Image Retrieval with Attentive Deep Local Features”. In: *IEEE/CVF ICCV*. 2017, pp. 3456–3465 (cit. on pp. 2, 9).

[51] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. “LF-Net: Learning Local Features from Images”. In: *NeurIPS*. 2018 (cit. on pp. 9, 10, 52).

[52] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. “Representation Learning with Contrastive Predictive Coding”. In: *arXiv preprint arXiv:1807.03748*. 2018 (cit. on pp. 25, 30, 50).

[53] World Health Organization. *Road traffic injuries*. 2022. URL: `https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries` (visited on 08/31/2022) (cit. on p. 1).

[54] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *IEEE/CVF CVPR*. 2017, pp. 652–660 (cit. on p. 11).

[55] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *NeurIPS*. 2017 (cit. on p. 11).

[56] René Ranftl and Vladlen Koltun. “Deep Fundamental Matrix Estimation”. In: *ECCV*. 2018, pp. 284–299 (cit. on p. 11).

[57] Krishna Regmi and Mubarak Shah. “Bridging the Domain Gap for Ground-to-Aerial Image Matching”. In: *IEEE/CVF ICCV*. 2019, pp. 470–479 (cit. on pp. 5–7).

[58] Tyler GR Reid, Sarah E Houts, Robert Cammarata, Graham Mills, Siddharth Agarwal, Ankit Vora, and Gaurav Pandey. “Localization Requirements for Autonomous Vehicles”. In: *SAE IJ-CAV*. 2019 (cit. on pp. 1, 17, 28, 47).

[59] Edward Rosten and Tom Drummond. “Machine Learning for High-Speed Corner Detection”. In: *ECCV*. 2006, pp. 430–443 (cit. on p. 10).

[60] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. “ORB: An Efficient Alternative to SIFT or SURF”. In: *IEEE/CVF ICCV*. 2011, pp. 2564–2571 (cit. on p. 10).

[61] Avishkar Saha, Oscar Mendez Maldonado, Chris Russell, and Richard Bowden. “Translating Images into Maps”. In: *IEEE ICRA*. 2021 (cit. on p. 13).

[62] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. “From Coarse to Fine: Robust Hierarchical Localization at Large Scale”. In: *IEEE/CVF CVPR*. 2019, pp. 12716–12725 (cit. on pp. 1–3, 9).

[63] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. “SuperGlue: Learning Feature Matching with Graph Neural Networks”. In: *IEEE/CVF CVPR*. 2020, pp. 4938–4947 (cit. on pp. 11, 52).

[64] Torsten Sattler, Michal Havlena, Konrad Schindler, and Marc Pollefeys. “Large-Scale Location Recognition and the Geometric Burstiness Problem”. In: *IEEE/CVF CVPR*. 2016, pp. 1582–1590 (cit. on p. 2).

[65] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. “Efficient & Effective Prioritized Matching for Large-scale Image-based Localization”. In: *IEEE TPAMI*. 2016, pp. 1744–1756 (cit. on p. 3).

[66] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. “The Graph Neural Network Model”. In: *IEEE TNN*. 2008, pp. 61–80 (cit. on p. 11).

[67] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “FaceNet: A Unified Embedding for Face Recognition and Clustering”. In: *IEEE/CVF CVPR*. 2015, pp. 815–823 (cit. on p. 25).

[68] Turgay Senlet and Ahmed Elgammal. “A Framework for Global Vehicle Localization Using Stereo Images and Satellite and Road Maps”. In: *IEEE/CVF ICCV Workshops*. 2011, pp. 2034–2041 (cit. on pp. 5, 12).

[69] Turgay Senlet and Ahmed Elgammal. “Satellite Image based Precise Robot Localization on Sidewalks”. In: *IEEE ICRA*. 2012, pp. 2647–2653 (cit. on p. 5).

[70] Xuelun Shen, Cheng Wang, Xin Li, Zenglei Yu, Jonathan Li, Chenglu Wen, Ming Cheng, and Zijian He. “RF-Net: An End-to-End Image Matching Network based on Receptive Field”. In: *IEEE/CVF CVPR*. 2019, pp. 8132–8140 (cit. on p. 9).

[71] Yujiao Shi, Dylan John Campbell, Xin Yu, and Hongdong Li. “Geometry-guided Street-view Panorama Synthesis from Satellite Imagery”. In: *IEEE TPAMI*. 2022 (cit. on p. 13).

[72] Yujiao Shi and Hongdong Li. “Beyond Cross-view Image Retrieval: Highly Accurate Vehicle Localization Using Satellite Image”. In: *IEEE/CVF CVPR*. 2022, pp. 17010–17020 (cit. on pp. 1–3, 5, 7–9, 17, 52).

[73] Yujiao Shi, Liu Liu, Xin Yu, and Hongdong Li. “Spatial-Aware Feature Aggregation for Cross-View Image based Geo-Localization”. In: *NeurIPS*. 2019, pp. 10090–10100 (cit. on pp. 2, 3, 5–7, 12, 16–18, 26).

[74] Yujiao Shi, Xin Yu, Dylan Campbell, and Hongdong Li. “Where am I looking at? Joint Location and Orientation Estimation by Cross-View Matching”. In: *IEEE/CVF CVPR*. 2020, pp. 4064–4072 (cit. on pp. 3, 7, 12, 16).

[75] Yujiao Shi, Xin Yu, Liu Liu, Dylan Campbell, Piotr Koniusz, and Hongdong Li. “Accurate 3-DoF Camera Geo-Localization via Ground-to-Satellite Image Matching”. In: *arXiv preprint arXiv: 2203.14148*. 2022 (cit. on pp. 3, 6, 12).

[76] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *arXiv preprint arXiv:1409.1556*. 2014 (cit. on pp. 6, 17, 18, 22, 26, 29).

[77] Linus Svärm, Olof Enqvist, Fredrik Kahl, and Magnus Oskarsson. “City-scale Localization for Cameras with Known Vertical Direction”. In: *IEEE TPAMI*. 2016, pp. 1455–1461 (cit. on p. 3).

[78] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005 (cit. on p. 1).

[79] Aysim Toker, Qunjie Zhou, Maxim Maximov, and Laura Leal-Taixé. “Coming Down to Earth: Satellite-to-Street View Synthesis for Geo-Localization”. In: *IEEE/CVF CVPR*. 2021, pp. 6488–6497 (cit. on pp. 5–7, 12).

[80] Engin Tola, Vincent Lepetit, and Pascal Fua. “DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo”. In: *IEEE TPAMI*. 2009, pp. 815–830 (cit. on p. 10).

[81] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. “Attention Is All You Need”. In: *NeurIPS*. 2017 (cit. on pp. 7, 44).

[82] Sebastiano Verde, Thiago Resek, Simone Milani, and Anderson Rocha. “Ground-to-Aerial Viewpoint Localization via Landmark Graphs Matching”. In: *IEEE SPL*. 2020, pp. 1490–1494 (cit. on pp. 2, 7, 8).

[83] Yannick Verdie, Kwang Yi, Pascal Fua, and Vincent Lepetit. “Tilde: A Temporally Invariant Learned Detector”. In: *IEEE/CVF CVPR*. 2015, pp. 5279–5288 (cit. on p. 10).

[84] Anirudh Viswanathan, Bernardo R Pires, and Daniel Huber. “Vision based Robot Localization by Ground to Satellite Matching in GPS-denied Situations”. In: *IEEE/RSJ IROS*. 2014, pp. 192–198 (cit. on pp. 5, 12).

[85] Nam N Vo and James Hays. “Localizing and Orienting Street Views using Overhead Imagery”. In: *ECCV*. 2016, pp. 494–509 (cit. on p. 2).

[86] Tingyu Wang, Zhedong Zheng, Chenggang Yan, Jiyong Zhang, Yaoqi Sun, Bolun Zheng, and Yi Yang. “Each Part Matters: Local Patterns Facilitate Cross-View Geo-Localization”. In: *IEEE TCSVT*. 2021 (cit. on pp. 3, 12, 13).

[87] Daniel Wilson, Xiaohan Zhang, Waqas Sultani, and Safwan Wshah. “Visual and Object Geo-localization: A Comprehensive Survey”. In: *arXiv preprint arXiv:2112.15202*. 2021 (cit. on p. 6).

[88] Scott Workman and Nathan Jacobs. “On the Location Dependence of Convolutional Neural Network Features”. In: *IEEE/CVF CVPR Workshops*. 2015, pp. 70–78 (cit. on pp. 2, 5, 6).

[89] Scott Workman, Richard Souvenir, and Nathan Jacobs. “Wide-Area Image Geolocalization with Aerial Reference Imagery”. In: *IEEE/CVF ICCV*. 2015, pp. 3961–3969 (cit. on pp. 5, 6).

[90] Zimin Xia, Olaf Booij, Marco Manfredi, and Julian FP Kooij. “Cross-View Matching for Vehicle Localization by Learning Geographically Local Representations”. In: *IEEE RA-L*. 2021, pp. 5921–5928 (cit. on pp. 1, 6, 9, 52).

[91] Zimin Xia, Olaf Booij, Marco Manfredi, and Julian FP Kooij. “Geographically Local Representation Learning with a Spatial Prior for Visual Localization”. In: *ECCV Workshops*. 2020, pp. 557–573 (cit. on pp. 1, 6).

[92] Zimin Xia, Olaf Booij, Marco Manfredi, and Julian FP Kooij. “Visual Cross-View Metric Localization with Dense Uncertainty Estimates”. In: *arXiv preprint arXiv:2208.08519*. 2022 (cit. on pp. 1–3, 5, 7–9, 17, 18, 23, 25, 26, 29, 30, 42, 44, 47).

[93] Hongji Yang, Xiufan Lu, and Yingying Zhu. “Cross-view Geo-localization with Layer-to-Layer Transformer”. In: *NeurIPS*. 2021 (cit. on pp. 5–7, 13).

[94] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. “LIFT: Learned Invariant Feature Transform”. In: *ECCV*. 2016, pp. 467–483 (cit. on pp. 9, 10).

[95] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. “Learning to Find Good Correspondences”. In: *IEEE/CVF CVPR*. 2018, pp. 2666–2674 (cit. on p. 11).

[96] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. “A Survey of Autonomous Driving: Common Practices and Emerging Technologies”. In: *IEEE Access*. 2020, pp. 58443–58469 (cit. on p. 1).

[97] Andi Zang, Xin Chen, and Goce Trajcevski. “High Definition Maps in Urban Context”. In: *Sigspatial Special*. 2018, pp. 15–20 (cit. on p. 1).

[98] Andi Zang, Zichen Li, David Doria, and Goce Trajcevski. “Accurate Vehicle Self-Localization in High Definition Map Dataset”. In: *ACM SIGSPATIAL Workshop*. 2017, pp. 1–8 (cit. on p. 1).

[99] Menghua Zhai, Zachary Bessinger, Scott Workman, and Nathan Jacobs. “Predicting Ground-Level Scene Layout from Aerial Imagery”. In: *IEEE/CVF CVPR*. 2017, pp. 867–875 (cit. on p. 7).

[100] Linguang Zhang and Szymon Rusinkiewicz. “Learning to Detect Features in Texture Images”. In: *IEEE/CVF CVPR*. 2018, pp. 6325–6333 (cit. on p. 10).

[101] Xu Zhang, Felix X Yu, Svebor Karaman, and Shih-Fu Chang. “Learning Discriminative and Transformation Covariant Local Feature Detectors”. In: *IEEE/CVF CVPR*. 2017, pp. 6818–6826 (cit. on p. 10).

[102] Chen Zhao, Zhiguo Cao, Chi Li, Xin Li, and Jiaqi Yang. “NM-Net: Mining Reliable Neighbors for Robust Feature Correspondences”. In: *IEEE/CVF CVPR*. 2019, pp. 215–224 (cit. on p. 11).

[103] Jianwei Zhao, Qiang Zhai, Rui Huang, and Hong Cheng. “Mutual Generative Transformer Learning for Cross-view Geo-localization”. In: *arXiv preprint arXiv:2203.09135*. 2022 (cit. on p. 7).

[104] Zhedong Zheng, Yunchao Wei, and Yi Yang. “University-1652: A Multi-view Multi-source Benchmark for Drone-based Geo-Localization”. In: *ACM Multimedia*. 2020, pp. 1395–1403 (cit. on p. 12).

[105] Sijie Zhu, Mubarak Shah, and Chen Chen. “TransGeo: Transformer Is All You Need for Cross-view Image Geo-localization”. In: *IEEE/CVF CVPR*. 2022, pp. 1162–1171 (cit. on p. 7).

[106] Sijie Zhu, Taojiannan Yang, and Chen Chen. “Revisiting Street-to-Aerial View Image Geo-localization and Orientation Estimation”. In: *IEEE/CVF WACV*. 2021, pp. 756–765 (cit. on pp. 3, 5, 6, 12, 16).

[107] Sijie Zhu, Taojiannan Yang, and Chen Chen. “VIGOR: Cross-View Image Geo-localization beyond One-to-one Retrieval”. In: *IEEE/CVF CVPR*. 2021, pp. 3640–3649 (cit. on pp. 1–9, 12, 16–21, 26–28, 32–36, 38–42, 45–47, 49, 51, 59–62).

A

Ground Truth Labels of VIGOR Dataset

We have visually inspected the image pairs of the VIGOR dataset and noticed a location inconsistency between image pairs that share the street image, as can be seen in Fig. A.1. Depending on the satellite image, the original ground truth location (yellow dot) is in different locations. However, this should be the same visual location for all satellite images belonging to this specific street image.

The authors of the VIGOR dataset [107] have used a ground resolution equal to 0.114 meters/pixel for all four cities of the dataset to convert the longitude and latitude of a street image to a satellite image location. This ground resolution is based on satellite images with a size of 640 x 640 pixels. We have measured the ground resolution ourselves, and it turns out that the ground resolutions differ per city. The measured value for the ground resolution of New York is almost equal to the ground resolution that the authors of the dataset have used. However, the resolutions of the other cities differ significantly (see Tab. A.1).

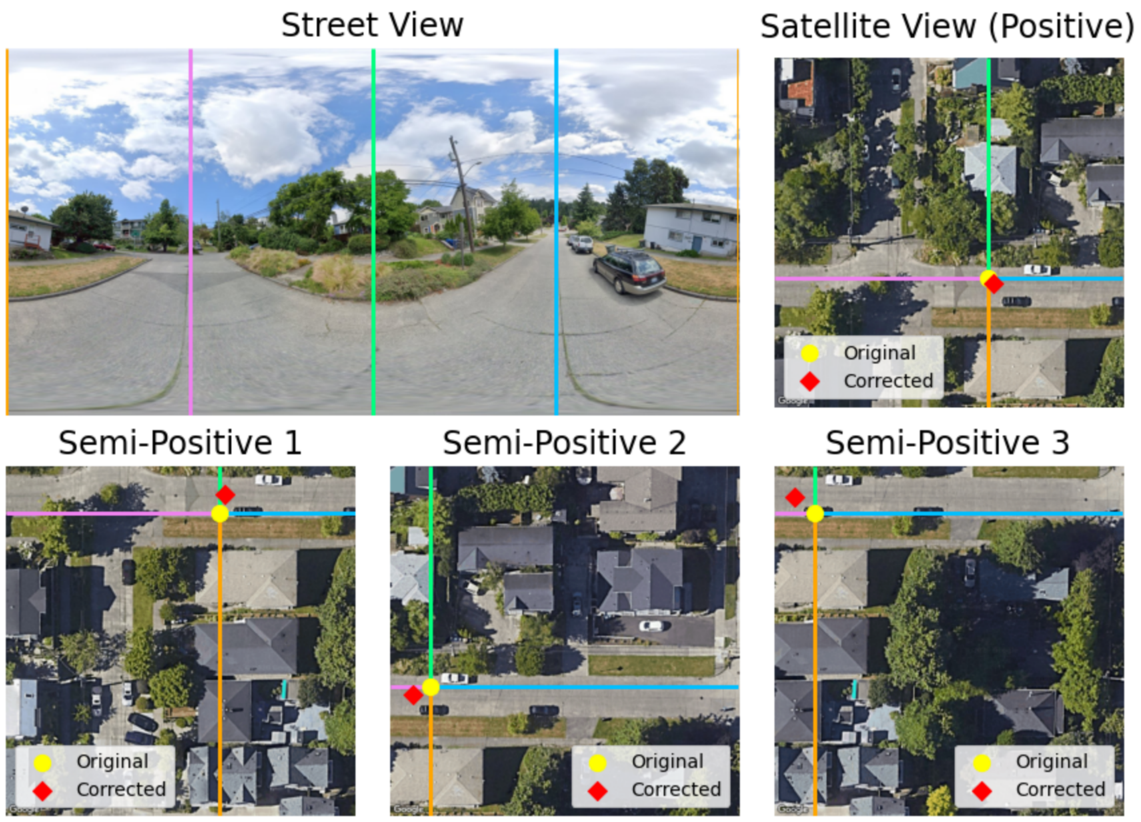


Figure A.1: A street image together with the four matching satellite images. South, West, North, and East are the orange, pink, green, and blue lines, respectively. The locations of the street image according to the original ground truth labels are indicated by the yellow dots. The red diamonds show the corrected street image locations. The images are part of the VIGOR dataset [107].

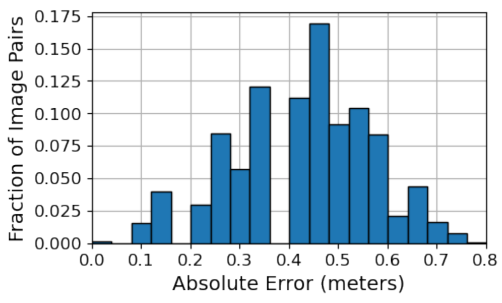
Table A.1: The original ground resolution and the measured ground resolutions for the four cities of the VIGOR dataset [107]. The original ground resolution was provided by the authors of the dataset, and the ground resolution for each city individually was measured by ourselves. The ground resolutions correspond to satellite images with a size of 640 x 640 pixels, and the unit of the ground resolution is meters/pixel.

Original ground resolution	Chicago	New York	San Francisco	Seattle
0.114	0.111	0.113	0.118	0.101

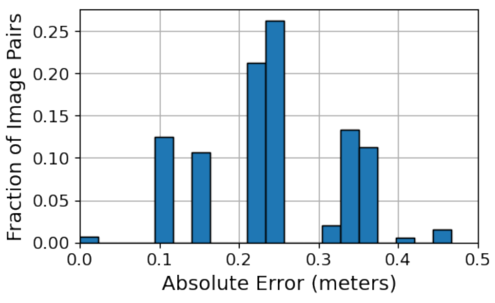
The difference between the original and the corrected labels has been determined. Tab. A.2 shows statistics on the absolute error, i.e., the distance between the original and the corrected location. The positive image pairs of the dataset were used to determine the statistics since only those image pairs were used for the experiments (see Section 4.1). Seattle has the highest mean absolute error because its ground resolution deviates the most from the original ground resolution (see Tab. A.1). Fig. A.2 shows histograms of the absolute error for the four cities.

Table A.2: Statistics on the absolute error of the original labels for the four cities of the VIGOR dataset [107]. The absolute error is defined as the distance between the original and the corrected locations. Only the positive image pairs of the dataset were used to determine the statistics.

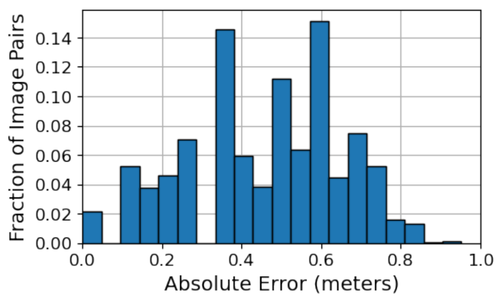
City	Absolute error (pixels)				Absolute error (meters)			
	Min.	Mean	Median	Max.	Min.	Mean	Median	Max.
Chicago	0.00	3.87	4.00	7.21	0.00	0.43	0.45	0.80
New York	0.00	2.18	2.24	4.14	0.00	0.25	0.25	0.47
San Francisco	0.00	3.90	4.12	8.06	0.00	0.46	0.49	0.95
Seattle	0.00	17.04	17.72	31.14	0.00	1.72	1.79	3.14



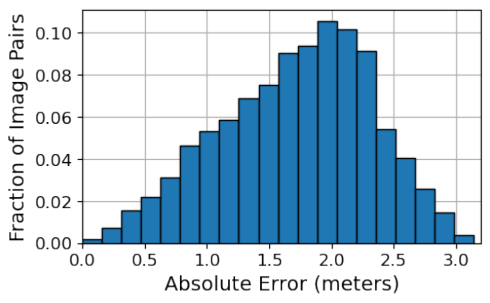
(a) Chicago



(b) New York



(c) San Francisco

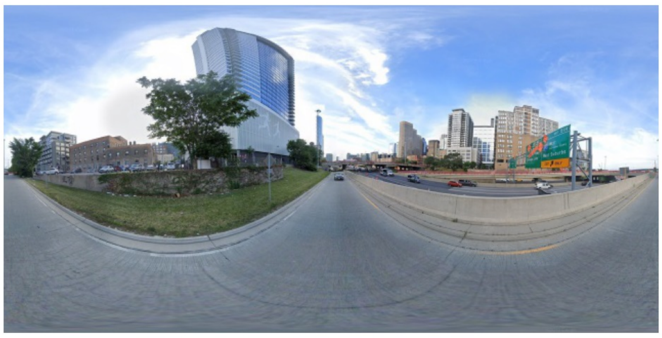


(d) Seattle

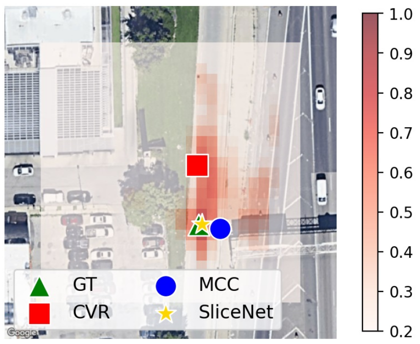
Figure A.2: Histograms of the absolute error of the original labels for the four cities of the VIGOR dataset [107]. The absolute error is defined as the distance between the original and the corrected locations. Only the positive image pairs of the dataset were used to determine the statistics.

B

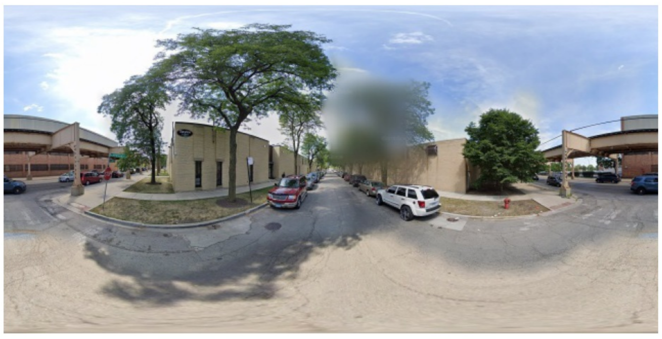
SliceNet Prediction Visualizations



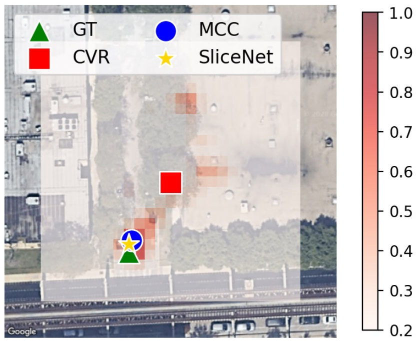
(a) Street image (image pair 13)



(b) Satellite image (image pair 13)

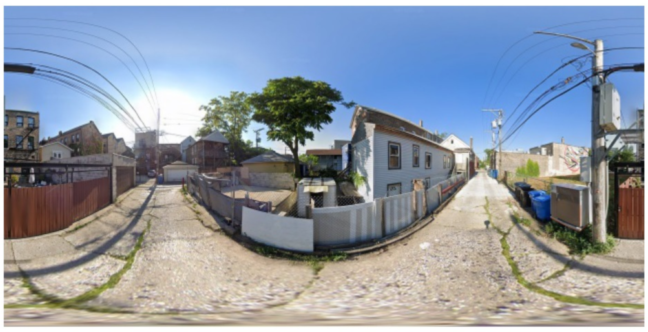


(c) Street image (image pair 14)

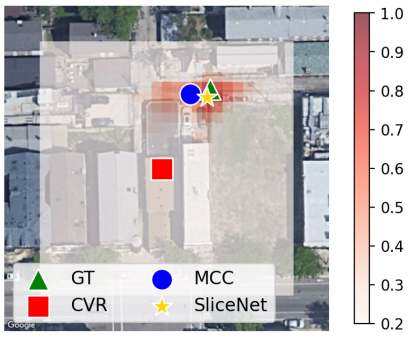


(d) Satellite image (image pair 14)

Figure B.1: Two image pairs with the corresponding similarity map in overlay on the satellite image. The model that has been used is SliceNet Select (matching map) with twenty horizontal slices and one vertical cell. Both image pairs are part of the same-area setting and are aligned. The images are part of the VIGOR dataset [107].



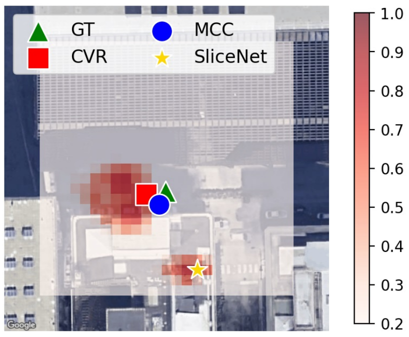
(a) Street image (image pair 15)



(b) Satellite image (image pair 15)

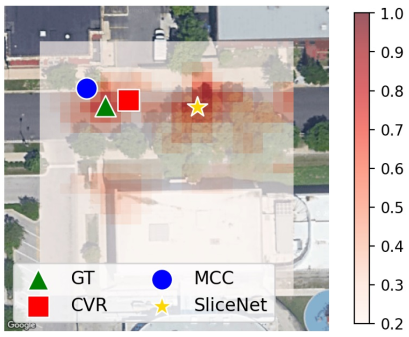
Figure B.2: One image pairs with the corresponding similarity map in overlay on the satellite image. The model that has been used is SliceNet Select (matching map) with twenty horizontal slices and one vertical cell. The image pair is part of the cross-area setting and is aligned. The images are part of the VIGOR dataset [107].

(a) Street image (image pair 16)

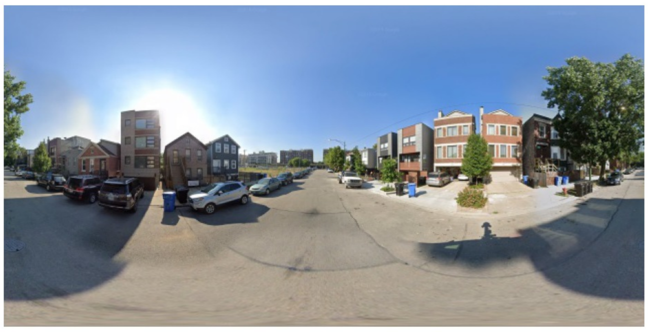


(b) Satellite image (image pair 16)

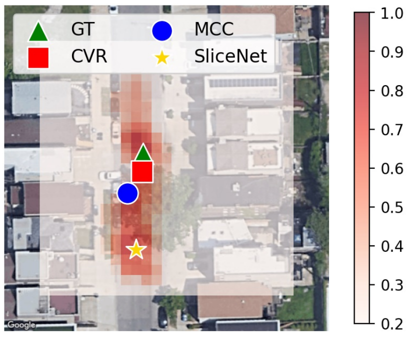
(c) Street image (image pair 17)



(d) Satellite image (image pair 17)



(e) Street image (image pair 18)



(f) Satellite image (image pair 18)

Figure B.3: Four image pairs with the corresponding similarity map in overlay on the satellite image. The model that has been used is SliceNet Select (matching map) with twenty horizontal slices and one vertical cell. The first image pair is part of the same-area setting and the last two image pairs are part of the cross-area setting. All image pairs are aligned. The images are part of the VIGOR dataset [107].