Delft University of Technology

Convolutional Cross-View Pose Estimation

Xia, Zimin; Booij, Olaf; Kooij, Julian F.P.

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Convolutional Cross-View Pose Estimation

Zimin Xia ©, Olaf Booij ©, and Julian F. P. Kooij ©, *Member, IEEE*

Abstract—We propose a novel end-to-end method for cross-view pose estimation. Given a ground-level query image and an aerial image that covers the query's local neighborhood, the 3 Degrees-of-Freedom camera pose of the query is estimated by matching its image descriptor to descriptors of local regions within the aerial image. The orientation-aware descriptors are obtained by using a translationally equivariant convolutional ground image encoder and contrastive learning. The Localization Decoder produces a dense probability distribution in a coarse-to-fine manner with a novel Localization Matching Upsampling module. A smaller Orientation Decoder produces a vector field to condition the orientation estimate on the localization. Our method is validated on the VIGOR and KITTI datasets, where it surpasses the state-of-the-art baseline by 72% and 36% in median localization error for comparable orientation estimation accuracy. The predicted probability distribution can represent localization ambiguity, and enables rejecting possible erroneous predictions. Without re-training, the model can infer on ground images with different field of views and utilize orientation priors if available. On the Oxford RobotCar dataset, our method can reliably estimate the ego-vehicle's pose over time, achieving a median localization error under 1 m and a median orientation error of around 1° at 14 FPS.

Index Terms—Aerial imagery, camera pose estimation, cross-view matching, localization, orientation estimation.

I. INTRODUCTION

LOCALIZATION is a core task in autonomous driving and outdoor robotics [1]. In urban canyons [2], Global Navigation Satellite System (GNSS), such as GPS, often has positioning errors of up to tens of meters due to the multipath effect. Thus, other sensors [3], such as camera [4], [5], [6] and LiDAR [7], [8], [9], are used in combination with detailed HD maps [10], [11] to enhance the localization accuracy and robustness. In practice, most commercial vehicles are not equipped with expensive LiDAR sensors. Besides, maintaining an up-to-date HD map is laborious and expensive, especially for areas in fast development. Hence, exploring alternative map sources for camera-based methods is an important and practical task. One promising map source is aerial imagery as it provides rich appearance information with global coverage.

We consider the task of cross-view camera pose estimation, namely, estimating the camera's location and orientation from a given ground-level query image by matching it to geo-referenced aerial imagery. Previous deep learning-based works [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25] successfully performed coarse city- or even country-scale localization, by formulating the localization as image retrieval, i.e. to find the aerial image from a reference database that contains the location of the ground query. More recently, there has been increasing interest in applying cross-view image retrieval for autonomous driving by zooming into a smaller geographically local region [26], [27], especially in the urban canyon where GNSS is prone to have large positioning errors. A few pioneer works [22], [28], [29], [30], [31], [32], [33] demonstrated the feasibility of pinpointing the 2D location, sometimes together with the orientation, of the ground camera within a known aerial image. Similar to [28], [29], [32], we are interested in the 3-Degrees-of-Freedom (3-DoF) camera pose, i.e. planar location and orientation (yaw), instead of the full 6-DoF pose, since the change in camera height, pitch, and roll are less important in downstream tasks in autonomous driving, such as motion prediction and planning.

However, several gaps must be filled before large-scale real-world deployment of cross-view camera pose estimation methods is a realistic possibility for self-driving. So far, the localization accuracy of existing methods is not yet good enough for autonomous driving requirements, e.g. the lateral and longitudinal error should be below 0.29 m [34]. Besides, many methods cannot be run at sufficiently low latency, i.e. ~15 frames per second (FPS), on datasets for self-driving [35], [36], [37]. For example, [28] relies on iterative optimization to estimate the ground camera's pose. In [33], computationally heavy Transformers are used to construct Birds Eye View (BEV) feature representations, and then the BEV representations from ground and aerial views are compared densely at each of the location-orientation combinations (i.e. 3-DoF poses). Both methods [28], [33] run at a low frame rate, e.g. 2 to 3 FPS. SliceMatch [32] fulfills the runtime requirement. However, it requires pre-computed slice masks for each possible pose it considers, which means that memory overhead limits the number of possible poses and therefore its accuracy. We also observe that when the aerial view contains a symmetric scene layout, e.g. at crossroads, single-mode regression-based methods [22], [31] might regress to a midpoint between visually similar locations, and optimization-based methods [28] might get stuck at a wrong local optimum.
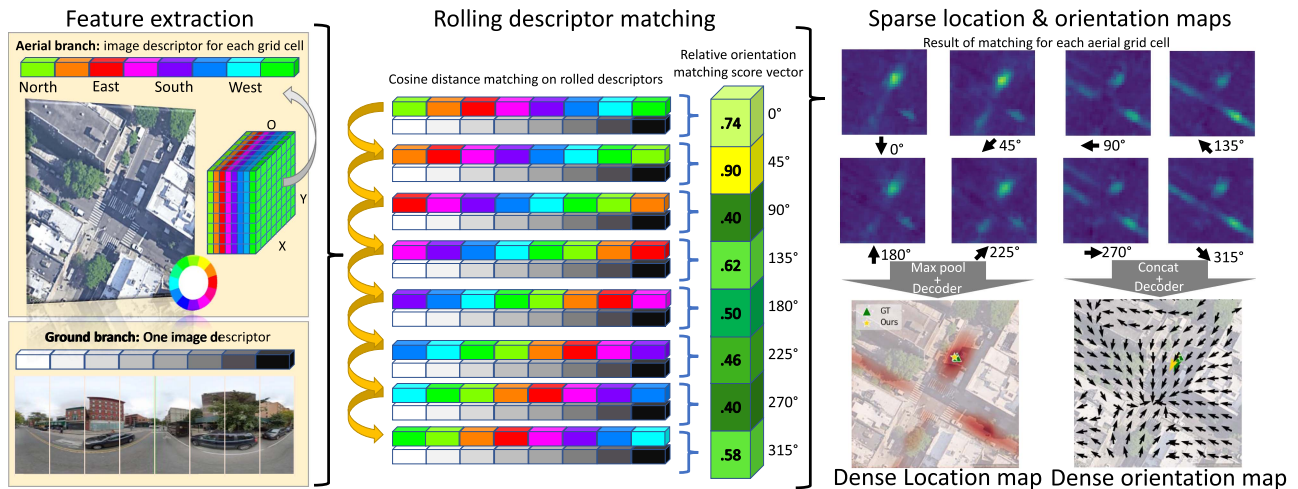
Fig. 1.   In Convolutional Cross-View Pose Estimation (CCVPE), ground and aerial images are encoded into orientation-aware image descriptors. For the aerial image, we create a grid of descriptors. Efficient joint localization and orientation prediction are enabled by matching rolled aerial descriptors with the ground descriptor. Sparse location and orientation maps are up-sampled into dense maps using decoders with coarse-to-fine matching. We max-pool descriptor matching scores over orientation channels to predict the most probable location considering different orientations. We concatenate the matching scores from different orientations to gather information for accurate orientation prediction. The final orientation prediction is conditioned on localization, i.e. it is selected at the predicted location in the dense orientation map.

To improve the pose estimation accuracy over prior works and meanwhile achieve fast runtime, we propose a novel method that predicts a multi-modal distribution for localization and jointly considers the orientation of the ground camera. As shown in Fig. 1, we exploit the translational equivariance property of convolutional networks to construct orientation-aware image descriptors that represent visual information in both ground and aerial views at different locations with a particular viewing direction. Joint localization and orientation estimation are achieved by convolving the ground descriptor on the aerial descriptors with circular padding, i.e. matching the ground descriptor to different rolled/shifted versions of the aerial descriptor. Then, our model regresses the fine-grained orientation based on discrete orientation matching scores and follows a coarse-to-fine formulation to gradually refine a sparse location map into dense output. The final output orientation is conditioned on the predicted location.

Our main contributions are: (i) We propose a novel method for end-to-end cross-view camera pose estimation, Convolutional Cross-View Pose Estimation (CCVPE).[1] It surpasses the previous state-of-the-art baselines by a large margin in localization and achieves comparable orientation estimation accuracy on VIGOR and KITTI datasets when testing generalization to new measurements within the same area and across different areas. (ii) CCVPE constructs a multi-modal distribution for localization and uniquely associates each location with its most probable orientation. We avoid a dense search over all 3-DoF poses (localization+orientation) by discretizing the orientation sparsely and performing additional regression. This formulation is efficient for fine-grained pose estimation. We also show the predicted probability can be used to filter out predictions that potentially have large localization or orientation errors. (iii) Our designed architecture exploits the strength of a translational

equivariant feature encoder and contrastive learning. Our ground image encoder maintains the spatial scene layout information relative to camera's viewing direction in the ground image descriptor and the contrastive loss enforces aerial descriptors to encode global orientation information. These descriptors enable jointly localization and orientation estimation with negligible extra computational cost. Without re-training, our model can infer the camera pose on images with different horizontal Field of Views (FoVs). In addition, it can utilize a coarse orientation prior, if available, to improve the localization without re-training.

An earlier version of our method was presented as a conference paper [29]. This article extends [29] by the following. First, we extended our method for end-to-end orientation estimation and jointly consider both location and orientation. Second, we re-designed our ground and aerial descriptors matching module and exploit coarse-to-fine matching. Third, we extended our experiments to include the recent KITTI cross-view localization dataset [28] and compared our method to additional recent state-of-the-art cross-view camera pose estimation baselines [28], [32]. Fourth, we demonstrated that our method can estimate the ego-vehicle pose at 14 FPS on the Oxford RobotCar dataset with a median lateral and longitudinal error below 1 m, and a median orientation error around $1°$.

II. RELATED WORK

In this section, we review the work related to cross-view camera pose estimation.

Cross-view image retrieval has shown great progress in the past years. It enjoys the advantage of the widely available geo-referenced aerial images and aims for rough geo-localization by retrieving the aerial image patch that covers the location of the ground-level query image. The first deep networks for this task date back to 2015 [12], [13], [14]. Since then, the common practice of using Siamese-like architecture was established. The

[1]Code is available at https://github.com/tudelft-iv/CCVPE

ground and aerial images are encoded into image descriptors by two network branches. Usually, these branches do not share weights [15], [16], [38], because two input images are from different domains. This domain gap is also one of the main challenges in the cross-view setting. Subsequent works seek to bridge the domain gap between the learned ground and aerial representations via various approaches.

An effective way for minimizing the domain gap is to construct visually similar inputs [15], [17], [21], [39], [40]. SAFA [15] observes that the polar rays in the aerial image correspond to the vertical lines in the ground image, and proposes to use a polar transformation on the aerial image to build an image that is visually similar to the ground view. In [40], an inverse polar transformation is used on ground-level panoramas to generate synthetic aerial images. In [17], the authors bridge the domain gap between the ground and aerial images by generating synthetic aerial images using GANs [41]. In [21], ground-level images are generated from aerial images using GANs, and the features for image generation are shared for cross-view image retrieval.

Besides constructing visually similar inputs, several works try to optimize the learned image feature for retrieval in different ways. CVM-Net [16] adopts the powerful image descriptor, NetVLAD [42], to learn how to gather local image features for building global image descriptors. In [38], the authors propose to use the orientation information of both views to guide the model to find more discriminative features across views. In [26], a rough localization prior from GNSS is considered during training to encourage the model to learn geographically locally discriminative features. CVFT [19] considers Optimal Transport theory to facilitate the feature alignment between ground and aerial images. Global-assists-local [25] addresses the case of retrieving a ground-level query with a limited horizontal FoV and proposes to embed the aerial feature outside the query's FoV into the aerial descriptor to aid the retrieval. In [43], the feature locality is explicitly enforced when building global image descriptors by partitioning the encoded features. CVLNet [44] gathers temporal information into the ground descriptor by making use of a ground-level query video. Recently, transformers are also used. L2LTR [23] introduces self-cross attention to flow effective information into the descriptors. TransGeo [24] proposes an attention-guided non-uniform cropping method to attend to and zoom in the informative local image patches. Apart from retrieval, several works also estimate the orientation of the ground camera [20], [45], [46], [47].

However, the major limitation of cross-view image retrieval is that the ground query is assumed to be located at the center of the matched aerial image patch, but we may not have aerial image patches whose center is in fact at the unknown test location. Densifying reference aerial patches reduces the influence of this assumption but increases the computation cost. In [39], the authors propose to zoom into the initial retrieved aerial image and crop smaller aerial patches at a set of candidate locations in the initial retrieved image for second-stage retrieval. A few works [27], [48], [49], [50] fuse the image retrieval results with temporal filters for more accurate localization. Still, estimating the accurate location and orientation of a single frame

ground-level query within a reference aerial image patch that covers it remains an open yet important task.

Cross-view camera pose estimation can be seen as a follow-up task after image retrieval or other coarse localization techniques. Given a ground-level query and an aerial image that covers the local surrounding of the query, the objective is to estimate the exact location and the orientation of the query within the given aerial image. In [22], a large-scale dataset for this task is introduced, and the authors propose a model that first retrieves an aerial image given the ground query with a known orientation and then regresses the location offset between them. Later, [31] also formulates the localization as a regression problem and includes an additional road extraction training objective. In [51], the orientation of the ground camera is estimated by assuming the location of the ground camera in the aerial image is known. Instead of regression, [28] solves the query ground camera pose by iterative optimization. It first warps the feature from the aerial image to a ground view using homography and then uses a multi-level Levenberg-Marquardt algorithm to estimate the 3-DoF ground camera pose using the warped aerial feature and extracted ground-level feature. SliceMatch [32] generates aerial descriptors at a set of candidate ground camera poses by pooling aerial features within the geometric extent of the viewing frustum of each pose. Then the ground descriptor is compared to all aerial descriptors for pose estimation. Vision Transformers [52] are used in [33] to map the features of the ground-level surrounding views to BEV, and the mapped BEV feature maps are densely compared to feature maps extracted from the aerial image for pose estimation. In [30], LiDAR measurements are fused with camera images for cross-view pose estimation.

However, there are several limitations in the above methods. Some methods only estimate the location [22], [31] or orientation [51] of the ground camera. Current regression [22], [31] or optimization [28], [30] formulation for localization restricts the output to a single mode without uncertainty estimation. When there are several visually similar locations in the aerial view, regression-based methods [22], [31] might regress to the midpoint between those locations, and optimization-based methods [28] might converge to a wrong local optimum. More importantly, these methods lack uncertainty estimation to reflect the quality of the outputs. Besides, the runtime is also a bottleneck in many existing methods, e.g. 2 to 3 FPS in [28], [30], [33]. SliceMatch [32] has fast runtime but it constructs descriptors only at the deepest level in the network and might miss fine-grained details for more accurate pose estimation. Besides, it needs pre-computed slice masks for aggregating features in different orientations. The memory overhead brought by the masks limits the number of tested poses and thus accuracy.

Floor map-based localization is a related task to cross-view camera pose estimation. Instead of aerial images, the goal is indoor localization of sensor measurements, e.g. camera images, in a building's BEV floor map of walls and rooms. LaLaLoc [53] renders ground-view floor layouts from a BEV floor map, and learns a shared descriptor space for query images and rendered layouts for end-to-end retrieval and pose refinement. LaLaLoc++ [54] removes the need for the rendering step in

LaLaLoc [53] and uses a UNet-like architecture [55] to build a descriptor at each candidate location. Localization is achieved by looking for locations whose local map descriptor is similar to the descriptor of the query. Laser [56] renders ground descriptors from a floor plan in an efficient way and formulates localization as metric learning. The above methods have many properties in common with cross-view camera pose estimation works. In particular, our method is similar to Laser [56] in constructing orientation-aware image descriptors for orientation estimation. However, Laser [56] cannot directly be used for cross-view camera pose estimation because it relies on explicit map occupancy boundary information, whereas the cross-view visual relation between outdoor ground and aerial images is not explicitly modeled but has to be learned.

III. METHODOLOGY

Given a ground-level color image G of size $H \times W \times 3$ and an aerial color image A of size $L \times L \times 3$ that covers the local surrounding of G, we aim to estimate the 3 Degrees-of-Freedom (DoF) pose, $\hat{\mathbf{P}} \in \mathbb{R}^2 \times \mathbb{SO}(2)$, of the camera that took G. Specifically, $\hat{\mathbf{P}} = [\hat{x}, \hat{o}]$. $\hat{x} = (\hat{u}, \hat{v})$ denotes the image coordinates of the location of the camera of G in the aerial image A. $\hat{o} \in [0°, 360°)$ denotes the orientation of the camera in the 2D aerial image plane: $0°$ means heading North, i.e. the up direction in the aerial image, and the orientation angle increases in the clockwise direction. Similar to other cross-view camera pose estimation methods [28], [32], we assume the pitch and roll angle of the ground camera are small, which is often the case for a vehicle-mounted camera.

A. Methodological Design Considerations

Existing cross-view camera pose estimation methods [22], [29], [31] use a Siamese network with two image encoders without weight-sharing, fuse the encoder's descriptors at the bottleneck, and finally, have a decoder provide the output. Our model follows a similar approach with a few novel modifications.

1. Multi-modal prediction: Instead of treating localization as a uni-modal estimation problem [22], [31], we propose to predict location with a discrete probability distribution D over the pixels in the $L \times L$ aerial image A, and formulate the learning as multi-class classification. This way, the output can capture the potential multi-modal localization ambiguity, and assign high probability to multiple distinct aerial locations that match the observed ground image G. The probabilistic output could be provided to a downstream robot localization stack for fusion with other sensors, or the *Maximum A-Posteriori* (MAP) location can be taken as a single localization estimate. Furthermore, the probability provides a confidence estimate suitable to reject unreliable predictions, as our experiments will demonstrate.

2. Coarse-to-fine descriptor matching: To obtain a high-resolution localization distribution, we propose to *match* a single ground descriptor to local regions in the aerial feature map, e.g. using the cosine similarity. The concept of learning a shared feature space where descriptors from different views are compared is also encountered in cross-view image retrieval [15], [16], [24], but we apply it for dense localization prediction. Our approach can therefore benefit from the contrastive learning loss to learn discriminative feature spaces for matching.

Furthermore, we observe that the discriminative visual information that distinguishes one aerial region from another depends on the aerial resolution and scale. We therefore propose to apply this descriptor matching approach in a coarse-to-fine manner, starting at the low-resolution bottleneck, doubling the feature map resolution each time until the full target resolution is reached. At each subsequent level, our approach will match the ground and aerial information and use the resulting matching score to guide the upsampling of the aerial feature to a higher spatial resolution. We will show that this improves localization accuracy.

3. Joint location and orientation matching: Location and orientation should be considered jointly. Estimating one, while ignoring the other could lead to sub-optimal estimation since the observed layout of the scene in the ground image G only relates to the BEV layout when both location and orientation of the ground camera are correct. Meanwhile, exploring a prior in one, e.g. orientation, should also benefit the estimation of the other, e.g. localization. This leads us to two considerations:

First, the image descriptors should *not* be invariant to different orientations. Instead, we construct ground descriptors where the elements correspond to information for specific viewing directions *relative* to the camera's unknown orientation, and aerial descriptors where dimensions capture information in specific *global* viewing directions (see Fig. 1 left). An aerial descriptor should only match the ground descriptor if the locations are similar, and if the viewing directions are aligned. By constructing ground descriptors that are *equivariant* with the camera's viewing direction (i.e. the horizontal image direction), the correct global orientation of the ground camera can be found by reordering its descriptor's feature dimensions ('rolling' the descriptors, see Fig. 1 middle) to match the local aerial descriptor.

Second, in addition to the Localization Decoder, we add an Orientation Decoder that predicts orientation as a function of the predicted location, i.e. it predicts a 2D vector field Y over the aerial view that maps each aerial location to the ground camera's most probable orientation if it would be located there. For instance, if the ground image shows the camera oriented towards a crossing, the localization uncertainty in the aerial view could be spread across the streets approaching the crossing, and each location would suggest a different global orientation (see Fig. 1 right). Uncertainty in the localization output thus also captures uncertainty over the global orientation.

4. Generalize to different horizontal FoVs: We aim for a model that can be used to match panoramic ground images, as well as images with a limited horizontal FoV without re-training, and can be trained with images of different FoVs for data augmentation. Therefore, other than constructing descriptors with a fixed length, our ground descriptors have a flexible length that depends on the horizontal FoV of the ground image G.
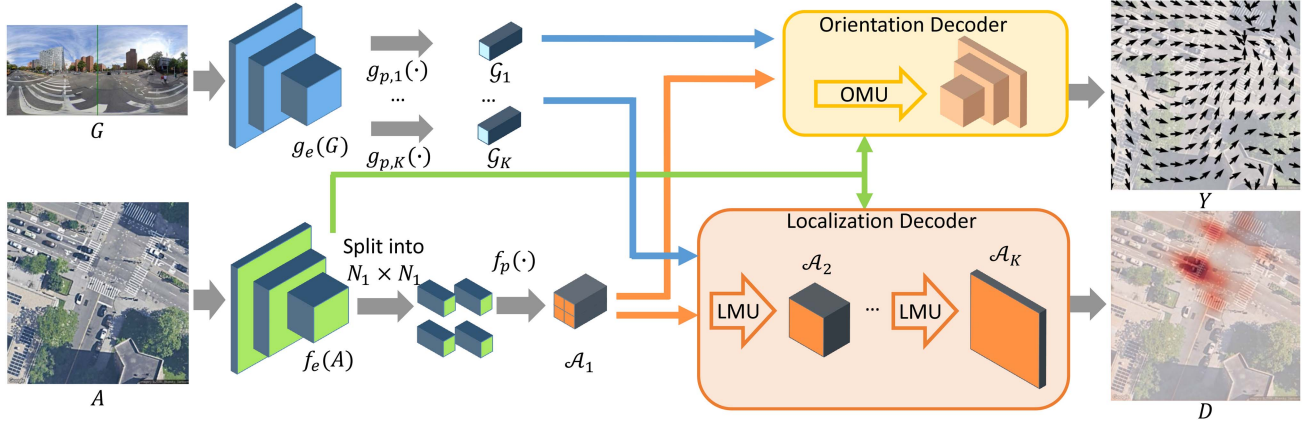
Fig. 2. Overview of our proposed Convolutional Cross-View Pose Estimation method, CCVPE. We overlay the output localization distribution (in red) and orientation vector field (black arrows) on top of the input aerial image for intuition.

B. Architecture Overview

The design considerations from Section III-A motivate our proposed Convolutional Cross-View Pose Estimation (CCVPE) architecture, shown in Fig. 2. One branch of the network, $g(\cdot)$, encodes the ground image G, and another branch, $f(\cdot)$, encodes the aerial image A. The descriptors from both encoders are matched in two specialized decoder branches: the Localization Decoder predicts the 2D spatial distribution D, the Orientation Decoder outputs the dense orientation vector field Y.

To match descriptors in a coarse-to-fine manner at K levels, the ground image G will be encoded into K ground descriptors $\mathcal{G}_k, k \in \{1, \ldots, K\}$, each of a different length C_k^G and capturing the relevant information to distinguish poses at that level's spatial resolution. Similarly, K aerial descriptor maps $\mathcal{A}_k$ are constructed to represent the relevant matching information of each local aerial region at level k. Each aerial descriptor $\mathcal{A}_k^{i,j}$ at spatial location (i,j) in the descriptor map $\mathcal{A}_k$ has a length of C_k^A, which represents all $360°$ viewing directions at that local region. When the ground descriptor $\mathcal{G}_k$ is encoded from a $360°$ panoramic ground image, it similarly has $C_k^G = C_k^A$. If the ground image instead has a limited horizontal FoV, then $C_k^G < C_k^A$ and its descriptors will later be matched to only C_k^G of the C_k^A aerial descriptor dimensions. The spatial resolution of the aerial descriptor map at level k is $N_k \times N_k = 2N_{k-1} \times 2N_{k-1}$, where $N_1 \times N_1$ is the lowest resolution at the bottleneck, and $N_K \times N_K = L/2 \times L/2$ is the last matching level K before the final output. The first aerial descriptor map, $\mathcal{A}_1$, is shared between the Localization Decoder and the Orientation Decoder.

In the Localization Decoder, see Fig. 3, the ground and aerial descriptors are compared at multiple resolution levels in a coarse-to-fine manner with our novel Localization Matching Upsampling (LMU) module. In the Orientation Decoder, we employ a similar Orientation Matching Upsampling (OMU) module, though only once after the bottleneck (our experiments will demonstrate this decoder does not benefit from coarse-to-fine matching). Similar to UNet [55] and other models for dense prediction tasks [57], [58], [59], we furthermore add skip
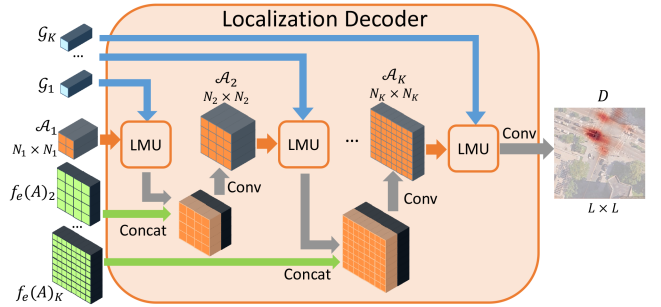
Fig. 3. Our proposed Localization Decoder.

connections from the aerial encoder to the two decoders between feature maps of same spatial resolution.

In the following, we provide details on our proposed descriptor construction, descriptor matching modules, localization and orientation decoders, and used loss functions.

C. Descriptors Construction

Both ground and aerial encoders $g(\cdot)$ and $f(\cdot)$ first apply their own feature extractor, $g_e(\cdot)$ and $f_e(\cdot)$, respectively. For the ground branch, we then use K ground feature projectors $g_{p,k}(\cdot), k \in \{1, \ldots, K\}$ to extract from the encoder's feature map the descriptors for the different coarse-to-fine levels, i.e. $\mathcal{G}_k = g_{p,k}(g_e(G))$. For the aerial branch, we split the aerial feature volume $f_e(A)$ into $N_1 \times N_1$ sub-volumes and use a shared aerial feature projector $f_p(\cdot)$ on each sub-volume $f_e(A)^{i,j}$ to generate the $N_1 \times N_1$ aerial descriptor map at level 1, $\mathcal{A}_1^{i,j} = f_p(f_e(A)^{i,j})$. The aerial descriptor maps $\mathcal{A}_k$ for $k > 1$ will be constructed within the Localization Decoder.

Our model will assume that ground images follow a cylindrical projection, namely that each column of pixels in the image represents the same number of degrees in the horizontal FoV. While cylindrical projections are commonly used for panoramic images, regular images with a limited horizontal FoV typically do not use this projection. We still model all ground images as

such, since we find this approximation still works well in practice[2]. We use a translational equivariant ground encoder $g(\cdot)$, therefore the length $C_k^G = C_k^A \times \frac{F}{360}$ of the ground descriptors $\mathcal{G}_k$ reflects the F degrees horizontal FoV of the ground image G.

Feature extractors: We use a regular convolutional network backbone as the ground and aerial feature extractors $g_e(\cdot)$ and $f_e(\cdot)$ on the input $H \times W \times 3$ ground image G and input $L \times L \times 3$ aerial image A, without sharing weights between these branches. We denote the shape of the encoded ground feature maps as $H' \times W' \times C'$, and of the aerial feature maps as $L' \times L' \times C'$.

Ground feature projector: A projector $g_{p,k}(\cdot)$ produces a single ground descriptor of length C_k^G. To reduce the computational cost of matching at increasingly higher spatial resolutions, the length of the ground descriptor at the next level is half of that of the ground descriptor at the current level, i.e. $C_k^G = 2\,C_{k+1}^G$.

Each projector consists of a 1×1 convolution to reduce the C' feature channels of the extracted ground feature $g_e(G)$ to $C_k' < C'$. To summarize the information along the vertical (height) direction in the scene while keeping it equivariant with the horizontal direction (relative viewing direction), we apply a fully-connected operation along the columns and squeeze the column dimension from H' to 1, resulting a $1 \times W' \times C_k'$ feature map. Finally, the ground descriptor $\mathcal{G}_k$ is created by reshaping this feature map into a 1D vector of length $C_k^G = W'C_k'$. These ground descriptors $\mathcal{G}_k$ are explicitly orientation-aware, as every block of C_k' elements captures the semantic content in a specific horizontal viewing direction relative to the camera's orientation.

Aerial feature projector: We create spatial granularity for localization by splitting the $L' \times L' \times C'$ aerial feature volume into $N_1 \times N_1$ feature sub-volumes. Then, a shared fully connected layer is used as our aerial feature projector $f_p(\cdot)$ to map each of the sub-volumes into an aerial descriptor $\mathcal{A}_1^{i,j}$. The orientation awareness of our aerial descriptors is encouraged by our loss function, which will align the aerial descriptors with the orientation-aware ground descriptors, see details in Section III-F.

D. Descriptor Matching Modules

To jointly consider location and orientation, we match ground descriptors at different locations in the aerial image and consider R different global orientations. In particular, the matching is done inside our proposed descriptor matching modules, the Localization Matching Upsampling (LMU) module, and the Orientation Matching Upsampling (OMU) module. As seen in Fig. 4, both modules rely on a Rolling & Matching strategy to compute descriptor matching scores.

Rolling & Matching: Both LMU and OMU use the ground descriptor $\mathcal{G}_k$ to 'match' the aerial descriptors $\mathcal{A}_k$ at each candidate location (i,j) with a defined global orientation $\frac{r}{R}360°$, $r \in \{1,\dots,R\}$, and output a feature volume with the higher spatial
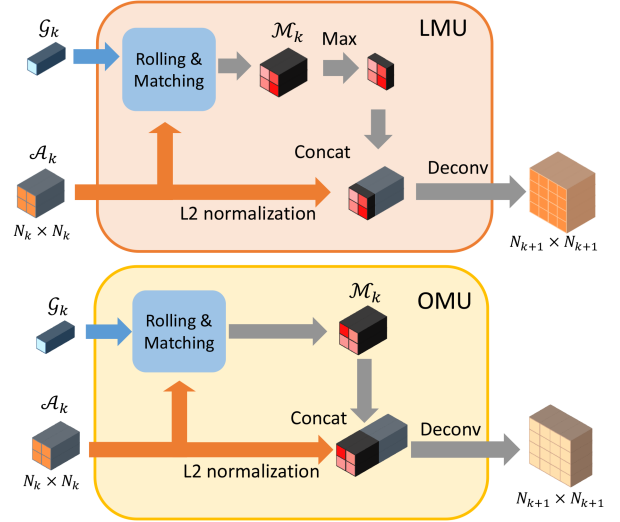

Fig. 4. Our proposed Localization Matching Upsampling (LMU) and Orientation Matching Upsampling (OMU) modules. Both generate an aerial feature map at a higher resolution than its input by matching the input aerial features to the ground features.

resolution of the next level $k+1$. To create R global orientations, $[0°, \frac{1}{R}360°, \dots, \frac{R-1}{R}360°]$, we 'roll' our orientation-aware aerial descriptor $\mathcal{A}_k^{i,j}$ at each candidate location (i,j) R times. Specifically, each 'rolling' is achieved by shifting all elements in $\mathcal{A}_k^{i,j}$ by a step length of $\frac{C_k^A}{R}$ to the front, and moving the C_k' front-most elements to the back. Note that we select R such that the rolling step length $\frac{C_k^A}{R}$ is a multiple of C_k'. The resulting aerial descriptors $\mathcal{A}_k^{i,j,r}$ each represents 'what the ground descriptor at level k should contain' at a particular location and global orientation combination $(i, j, \frac{r}{R}360°)$.

Matching each aerial descriptors $\mathcal{A}_k^{i,j,r}$ to the ground descriptor $\mathcal{G}_k$ is then done by the cosine similarity. Whereas each $\mathcal{A}_k^{i,j,r}$ captures the environment's appearance in all global directions with a C_k^A-dimensional vector, the C_k^G-dimensional ground descriptor $\mathcal{G}_k$ may represent images with a limited horizontal FoV, i.e. $C_k^G < C_k^A$. Therefore, we crop the middle C_k^G elements from $\mathcal{A}_k^{i,j,r}$, denoted as $\overline{\mathcal{A}}_k^{i,j,r}$, to match the same-sized descriptors of the same FoV using,

$$\mathcal{M}_k^{i,j,r} = \text{sim}(\mathcal{G}_k, \mathcal{A}_k^{i,j,r}) = \frac{\mathcal{G}_k \cdot \overline{\mathcal{A}}_k^{i,j,r}}{\|\mathcal{G}_k\|_2 \times \|\overline{\mathcal{A}}_k^{i,j,r}\|_2}. \quad (1)$$

The Rolling & Matching can be seen as convolving a kernel $\mathcal{G}_k$ over $\mathcal{A}_k^{i,j}$ with a stride of $\frac{C_k^A}{R}$, circular padding, and extra normalization. The resulting $N_k \times N_k \times R$ matching score volume $\mathcal{M}_k$ expresses how similar the ground descriptor is to the aerial descriptor at each candidate location and orientation.

LMU: The LMU summarizes the localization cues from $\mathcal{M}_k$ in an invariant manner to the different global orientations. We therefore take for each location the maximum matching score over the R orientations. These $N_k \times N_k \times 1$ max scores are concatenated to the L2-normalized $N_k \times N_k$ aerial descriptors

---

[2]See supporting experiments in the Supplementary Material.

to guide the upsampling of the aerial feature through a deconvolution. We show in our ablation study that the L2-normalization before feature concatenation is crucial for good pose estimation performance.

Notably, a prior in the ground camera's orientation is often available for vehicle localization, e.g. indicated by the driving direction. Incorporating such an orientation prior is straightforward in the LMU by removing the non-corresponding orientation channels in the matching score volume $\mathcal{M}_k$. This does not require any retraining.

OMU: Instead of extracting features that are orientation-invariant, OMU explicitly maintains the orientation information in $\mathcal{M}_k$. It has a similar design as LMU other than that the $\mathcal{M}_k$ is directly concatenated to the L2-normalized aerial descriptors. Thus the deconvolution layer can make use of information on how the ground descriptor $\mathcal{G}_k$ matches aerial descriptors $\mathcal{A}_k$ at all R orientations. We apply OMU only to matching level 1 in the Orientation Decoder (we explored other settings, but did not observe a clear benefit).

E. Decoders

We have two separate decoders for localization and orientation estimation.

Localization Decoder: Our Localization Decoder contains K LMU modules to gradually increase the spatial resolution of ground and aerial descriptor matching and finally generates a discrete distribution D over the pixels of $L \times L$ aerial image A for localization, see Fig. 3.

At each level k, the output feature from LMU is concatenated with the skip-connected aerial feature $f_e(A)_{k+1}$ of the same spatial resolution from the aerial feature extractor $f_e(\cdot)$ to access the scene layout information. Then, 2D convolution is applied to generate the aerial descriptors $\mathcal{A}_{k+1}$ for level $k+1$. After the LMU at level K, the output feature volume would have a spatial resolution $L \times L$, where $L = 2N_K$. Next, 2D convolution with a softmax activation is applied to convert the feature volume into a $L \times L \times 1$ discrete distribution D, in which the values denote how probable the ground camera is located at each pixel location (i, j). The *Maximum A-Posteriori* (MAP) pixel location $(\hat{i}, \hat{j})$ in D is taken as our final localization estimation, and the image coordinate of its center is our final prediction, $(\hat{u}, \hat{v}) = (u_{\hat{i}}, v_{\hat{j}})$.

Orientation Decoder: Our Orientation Decoder up-samples the coarse orientation information into a dense orientation vector field. It contains an OMU module at the beginning to match the ground descriptor $\mathcal{G}_1$ and aerial descriptors $\mathcal{A}_1^{i,j}$ at level 1 and upsamples the resulting matching score volume together with the L2-normalized aerial descriptors to spatial resolution $N_2 \times N_2$. The remainder of our Orientation Decoder uses a series of deconvolutions and convolutions to further upsample the feature volume to the target resolution $L \times L$. Similar to our Localization Decoder, there is a skip connection that passes aerial feature $f_e(A)_k$ from the aerial encoder to the Orientation Decoder. The final output of our Orientation Decoder is an $L \times L \times 2$ vector field Y denoting the predicted orientation at each pixel location (i, j) in the aerial image A. The feature channel is L2-normalized and we use the first and second channels to represent the cosine

and sine of the predicted orientation angle. The final orientation prediction $\hat{o}$ is selected in Y at the predicted pixel location $(\hat{i}, \hat{j})$, i.e. $\hat{o} = Y^{(\hat{i}, \hat{j})}$.

F. Loss Functions

Our loss $\mathcal{L}$ consists of three parts: a contrastive learning loss $\mathcal{L}_\mathcal{M}$, a classification loss $\mathcal{L}_D$ for localization, and a regression loss $\mathcal{L}_Y$ for orientation estimation. The ground truth location is represented by a discrete distribution D_{gt} of size $L \times L$. In practice, we place a 2D Gaussian distribution at the ground truth pixel coordinates (i_{gt}, j_{gt}) to form a smooth ground truth distribution D_{gt}.

The contrastive learning loss: $\mathcal{L}_\mathcal{M}$ is an average over contrastive learning losses $\mathcal{L}_{\mathcal{M}k}$, at K levels, i.e. $\mathcal{L}_\mathcal{M} = \frac{1}{K}\sum_{k=1}^{k=K} \mathcal{L}_{\mathcal{M}k}$. At each level k, $\mathcal{L}_{\mathcal{M}k}$ is applied on the matching score volume $\mathcal{M}_k$ to encourage the aerial descriptors for locations and orientations close to the ground truth poses to match the ground descriptor $\mathcal{G}_k$. Since our ground descriptors are orientation equivariant, training with $\mathcal{L}_{\mathcal{M}k}$ enforces the aerial descriptors at the correct locations to be orientation equivariant as well.

The location and orientation space of $\mathcal{M}_k$ is discretized into $N_k \times N_k \times R$, and the ground truth location and orientation would never exactly be centered at a grid point. We therefore express the closeness of indices (i, j, r) at level k to the true pose with weights $w_k^{i,j,r} = w_k^{i,j} \times w_k^r$. To obtain spatial weights $w_k^{i,j}$, we reduce the spatial dimensions of D_{gt} from $L \times L$ to $N_k \times N_k$ by max pooling. To compute weights w_k^r over the R orientation channels, we first find the orientation indices r_1 and r_2 of the discrete angles closest to the true orientation, $360° \times (r_1/R) < o_{gt} < 360° \times (r_2/R)$. We only assigning non-zero weight to r_1 and r_2, where their weight is inversely proportional to their relative angular distance to o_{gt}, and $w_k^{r_1} + w_k^{r_2} = 1$. We can now finally define $\mathcal{L}_{\mathcal{M}k}$ as a weighted sum $\mathcal{L}_{\mathcal{M}k} = \sum_{i,j,r} w_k^{i,j,r} \mathcal{L}'_{\mathcal{M}k}(i, j, r)$ of infoNCE losses [60] on the cosine similarity of (1),

$$\mathcal{L}'_{\mathcal{M}k}(i, j, r) = -\log \frac{\exp(\text{sim}(\mathcal{G}_k, \mathcal{A}_k^{i,j,r})/\tau)}{\sum_{i',j',r'} \exp(\text{sim}(\mathcal{G}_k, \mathcal{A}_k^{i',j',r'})/\tau)}. \quad (2)$$

The localization loss: We formulate the localization problem as a multi-class classification. In our main setting, the localization loss $\mathcal{L}_D$ is a cross-entropy loss,

$$\mathcal{L}_D = -\sum_{i=1}^{L}\sum_{j=1}^{L} D_{gt}^{i,j} \log D^{i,j}, \quad (3)$$

where (i, j) are pixel coordinates.

As an alternative, we also consider training the localization distribution D by minimizing the transported mass from D to D_{gt} based on Optimal Transport theory [61]. For this, we use a Wasserstein distance-based loss [62] as our $\mathcal{L}_D$. Unlike cross-entropy, Wasserstein distance considers the distance between the mass in the source and target distributions. To compute the Wasserstein distance loss $\mathcal{L}_D$ between D and D_{gt} efficiently, we

here define D_{gt} as a one-hot distribution. This loss then becomes,

$$\mathcal{L}_D = -\sum_{i=1}^{L}\sum_{j=1}^{L} d(i,j) \cdot D^{i,j}. \tag{4}$$

In (4), $d(i,j)$ denotes the L2-distance in pixels between the pixel location (i,j) in D to the ground truth pixel location (i_{gt}, j_{gt}), i.e. $d(i,j) = \sqrt{(i-i_{gt})^2 + (j-j_{gt})^2}$. We compare the cross-entropy loss and Wasserstein distance-based loss in our ablation study, but will use the cross-entropy loss as $\mathcal{L}_D$ in our main experiments.

The orientation loss: Instead of treating the orientation prediction as a discrete classification problem, which would result in a large number of classes for joint localization and orientation supervision, we formulate this problem as regression. Since we use a Gaussian smoothed ground truth D_{gt}, we sum the contributions from smoothed ground truth locations and define our orientation loss $\mathcal{L}_Y$ as,

$$\mathcal{L}_Y = \sum_{i=1}^{L}\sum_{j=1}^{L} D_{gt}^{i,j}\left(\left(\cos(o_{gt}) - Y_1^{i,j}\right)^2 + \left(\sin(o_{gt}) - Y_2^{i,j}\right)^2\right). \tag{5}$$

In (5), o_{gt} is the ground truth orientation, Y is the $L \times L \times 2$ predicted orientation vector field. Y_1 and Y_2 denote the first and second channel of Y. Multiplying with D_{gt} removes the contribution to the orientation loss $\mathcal{L}_Y$ at wrong locations.

The total loss: $\mathcal{L}$ is a weighted combination of the localization loss $\mathcal{L}_D$, orientation loss $\mathcal{L}_Y$, and contrastive learning loss $\mathcal{L}_M$:

$$\mathcal{L} = \mathcal{L}_D + \alpha\mathcal{L}_Y + \beta\mathcal{L}_M, \tag{6}$$

where the α and β are hyperparameters that weigh the importance of $\mathcal{L}_Y$ and $\mathcal{L}_M$ during training.

IV. EXPERIMENTS

In this section, we first introduce the three used datasets, followed by our evaluation metrics. After this, we provide our implementation details. Then, our CCVPE method is compared to previous state-of-the-art baselines w.r.t. generalization to new measurements within the same areas and across different areas. Next, we study how our method works with an orientation prior and on images with different horizontal FoVs. Then, we use our method to estimate the pose of the ego-vehicle along test traversals using sequences of ground images. Finally, an extensive ablation study and an analysis of runtime are provided.

A. Datasets

We test the generalization of all models to new measurements within the same areas and across different areas on the VIGOR [22] and KITTI [37] datasets. On the Oxford RobotCar dataset [35], [63], our method is used to estimate the ego-vehicle pose frame-by-frame using the ground image sequence collected in test traversals. The original KITTI and Oxford RobotCar datasets do not contain any aerial images, therefore we make use of the collected aerial images from [39] for KITTI, and [27], [29] for Oxford RobotCar.

The VIGOR dataset [22] contains ground-level panoramic images and aerial patches collected in four cities in the US. The aerial patches are distributed regularly as a grid, providing seamless coverage of the 4 target cities. Each aerial patch covers a $\sim$70 m $\times$ 70 m ground region. The orientation of the panorama and aerial patch is aligned such that the center vertical line in the panorama corresponds to the up direction (North) in the aerial patch. In our experiments, changing the orientation of the ground panorama is achieved by shifting the image along the horizontal axis. Reducing the horizontal FoV is achieved by dropping the image columns at the left and right borders. Since the ground truth labels are improved by [32], we use those more accurate labels. The VIGOR dataset defines the aerial patches as either positive or semi-positive for each ground image. An aerial patch is positive if its center $1/4$ region contains the ground camera's location, otherwise, it is semi-positive. In our experiments, we use positive aerial images for training and testing all models. We adopt the Same-Area and Cross-Area split from [22]. On the Same-Area split, we train models on images from all four cities and test models on images from the same cities. Training and test sets do not share any ground images but may share aerial patches. On the Cross-Area split, models are trained on image pairs from New York and Seattle and tested on pairs from Chicago and San Francisco. For validation and hyperparameter tuning, we randomly select 20% of the data from the training set.

The KITTI dataset [37] is collected by a vehicle platform in Karlsruhe, Germany, covering city, rural area, and highway scenarios. The stereo camera faces the driving direction and has a horizontal FoV of $90°$. In [28], the authors make use of the images from the left camera of the stereo camera and collected aerial images with ground resolution $\sim$0.20 m/pixel to enable cross-view pose estimation. Each aerial patch covers a $\sim$100 m $\times$ 100 m ground area. The data is split into Training, Test 1, and Test 2 sets. Images in Training and Test 1 sets are from the same regions. Images in Test 2 set are from different areas than those in the Training set. In our experiments, we refer to Test 1 and Test 2 sets as Same-Area and Cross-Area. As assumed in [28], ground images are located within a 40 m $\times$ 40 m area in the corresponding aerial patches' center, and there is an orientation prior with noise between $-10°$ and $10°$. In this case, a random rotation between $-10°$ and $10°$ is applied on each aerial image whose 'East' orientation was aligned with the ground image. In our experiments, we adopt the same setting, and also provide extra results for unknown orientations where a random rotation from the $360°$ circular domain is applied on each orientation-aligned aerial image.

The Oxford RobotCar dataset [35], [63] contains videos with a limited horizontal FoV collected over multiple traversals at different times, seasons, and weather conditions, along the same route in Oxford, U.K. In [26], [27], the authors collected aerial patches for retrieval, and later [29] stitched those aerial patches with their collected extra ones into a continuous aerial image that covers the Oxford area. We follow the same setting as in [27], [29] that the training, validation, and test data are from different traversals to test our model's generalization to different dynamic objects, weather, and lighting conditions across time. Instead

of directly using the sparse test images used in [27], [29], we sample test ground images from the original Oxford RobotCar dataset [35], [63] at a higher frame rate, $\sim$1.6 FPS, for our experiment of ego-vehicle following. In total, there are three test traversals, enabling testing in Summer and Winter. During training, aerial patches that cover $\sim$74 m $\times$ 74 m ground area are randomly cropped from the continuous map around a location that is less than $\sim$26 m away from the vehicle's location. For validation and testing, we use the same set of aerial patches as in [29]. In our experiment, the orientation of the ground camera is always assumed unknown, so we simply use the ground images and north-aligned aerial images as input pairs.

B. Baselines Methods

We compare CCVPE to two types of baselines.

First, we include the state-of-the-art (SOTA) cross-view pose estimation baselines: the cross-view regression method (CVR) [22], iterative optimization method LM [28], and Slice-Match [32]. CVR [22] is originally designed for joint image retrieval and location regression. For a fair comparison, we train it for localization within a given aerial image (we find it achieves better localization error than training it for retrieval + localization). We also trained a CVR model using the same EfficientNet-B0 [64] as its feature extractor, denoted as Eff-CVR. LM [28] uses an iterative method to estimate the location and orientation of the ground camera on the aerial image. We use the provided model by its author and the same setting on its prior [28] on KITTI dataset, namely, the ground images are located in a 40 m $\times$ 40 m area in the input aerial image center, and a rough orientation prior with noise between $-10°$ and $10°$ is available during training and test time. For completeness, we also include the model trained and tested without any orientation prior.

On the KITTI dataset, [39] evaluated several image retrieval or retrieval with orientation estimation baselines by limiting their searching area to a region of 40 m $\times$ 40 m. We include the same fine-grained cross-view image retrieval baselines from [39].

C. Evaluation Metrics

We use mean and median error over all test samples as our main evaluation metrics for both localization and orientation prediction. For localization, the error is the distance in meters between the predicted location and the ground truth location. For orientation, the error is the angular difference (°) between the predicted camera orientation at the predicted location and ground truth camera orientation. In addition, we report the percentage of test samples that has an error below certain thresholds, namely 1 m, 3 m, and 5 m for localization, and $1°$, $3°$, and $5°$ for orientation. For localization, longitudinal and lateral error w.r.t. the vehicle's driving direction are given separately [28]. For image retrieval methods, the localization error is calculated by measuring the distance between the center of the retrieved aerial image patch and the ground truth location. To measure if the true location receives probability mass, and thus would not be discarded if used in a probabilistic temporal filter, we also measure the predicted probability at the ground truth pixel.

For the baseline method that regresses [22] to a single location without uncertainty estimates, we assume their prediction is the peak of an isotropic Gaussian distribution, and estimate the standard deviation of this Gaussian distribution on the validation set. SliceMatch [32] measures descriptors' similarity scores on their candidate poses. We use the scores at the candidate location that is closest to the ground truth location to derive the predicted probability at the ground truth pixel. Finally, we measure our model's runtime on a Tesla V100 GPU.

D. Implementation Details

EfficientNet-B0 [64] is used as our ground and aerial feature extractors, $g_e(\cdot)$ and $f_e(\cdot)$. There is no weight-sharing between them. When the ground image G is panoramic, circular padding in the horizontal direction is used inside the ground encoder $g(\cdot)$, and zero padding otherwise. For other model components, and the vertical direction padding in $g(\cdot)$, we use zero padding. During training, the feature extractor is initialized from ImageNet [65] pre-trained weights, and other components are initialized randomly. Our model is trained using the Adam optimizer [66] with a learning rate of 0.0001. We use the default drop connect [67] rate, 0.2, from EfficientNet [64], and the default $\tau = 0.1$ in (2), from infoNCE loss [60]. Different weights $\alpha = 1 \times 10^{-2}, \ldots, 1 \times 10^2$ and $\beta = 10, \ldots, 1 \times 10^5$ were tested for weighing the orientation loss $\mathcal{L}_Y$ and contrastive learning loss $\mathcal{L}_M$, and $\alpha = 10$ and $\beta = 1 \times 10^4$ are selected since they provide best validation performance. The model bottleneck size $N_1 \times N_1$ is set to 8×8, and consequently there are $K = 6$ levels in our coarse-to-fine descriptor matching.

Note that, even though our method assumes ground images follow cylindrical projection, for KITTI and Oxford RobotCar datasets we directly input their perspective images. Our detailed study in the Supplementary Material shows that both projections work equally well in practice.

E. Generalization to New Measurements in Same Area

First, we compare our method, CCVPE, to baselines for generalizing to new measurements (i.e. ground images) in the same area. This corresponds to use cases that target operation in a predetermined area, such as driving in one city, so models can be trained on data from that specific area. For this task, we report the evaluation results on VIGOR Same-Area test set and KITTI Same-Area (Test 1) set.

Pose estimation on VIGOR Same-Area: As shown in Table I Same-Area, when testing with images with known orientation, our method surpasses all baselines, CVR [22], Eff-CVR, and SliceMatch [32], w.r.t. mean and median localization errors. Replacing the VGG [68] backbone with EfficientNet-B0 [64] for CVR improves localization performance, but Eff-CVR still has significantly higher localization errors than ours. When the orientation of test images is unknown, our method beats the previous SOTA SliceMatch by a large margin in localization, i.e. 56% in the mean error and 72% in the median error. Regarding orientation estimation, CVR could not infer the orientation of

TABLE I
EVALUATION ON VIGOR SAME-AREA AND CROSS-AREA TEST SET

VIGOR test		Same-Area						Cross-Area					
		↓ Localization (m)		↓ Orientation (°)		↑ P@GT (10⁻³)		↓ Localization (m)		↓ Orientation (°)		↑ P@GT (10⁻³)	
		mean	median	mean	median	mean	median	mean	median	mean	median	mean	median
0°	CVR [22]	8.82	7.68	-	-	0.02	0.02	9.45	8.33	-	-	0.02	0.02
	Eff-CVR	7.89	6.25	-	-	0.02	0.03	8.27	6.60	-	-	0.02	0.03
	SliceMatch [32]	5.18	2.58	-	-	0.06	0.05	5.53	2.55	-	-	0.06	0.06
	CCVPE (ours)	**3.60**	**1.36**	-	-	**1.60**	**1.12**	**4.97**	**1.68**	-	-	**1.08**	**0.71**
360°	SliceMatch [32]	8.41	5.07	28.43	**5.15**	0.02	0.02	8.48	5.64	**26.20**	**5.18**	0.02	0.02
	CCVPE (ours)	**3.74**	**1.42**	**12.83**	6.62	**1.47**	**1.00**	**5.41**	**1.89**	27.78	13.58	**0.93**	**0.58**

Best in bold. We report mean and median localization and orientation error, as well as the probability at the ground truth pixel in the aerial image, denoted as 'P@GT'. The leftmost column indicates orientation uncertainty: '0°' means testing with known orientation, such that the center vertical line in the panorama corresponds to the North direction in the aerial image, and the known orientation is used to remove the non-corresponding orientation channels in the matching score volume; '360°' means the test orientation is unknown and then the panoramic ground image was horizontally shifted corresponding to a random angle in the 360° circular domain.

TABLE II
EVALUATION ON KITTI DATASET

Same-Area		↓ Localization (m)		↑ Lateral (%)			↑ Longitudinal (%)			↓ Orientation (°)		↑ Orientation (%)		
		mean	median	1m	3m	5m	1m	3m	5m	mean	median	1°	3°	5°
retrieval	CVM-Net [16]	-	-	5.83	17.41	28.78	3.47	11.18	18.42	-	-	-	-	-
	CVFT [19]	-	-	7.71	22.37	36.28	3.82	11.48	18.63	-	-	-	-	-
	SAFA [15]	-	-	9.49	29.31	46.44	4.35	12.46	21.10	-	-	-	-	-
	Polar-SAFA [15]	-	-	9.57	30.08	45.83	4.56	13.01	21.12	-	-	-	-	-
	DSM [47]	-	-	10.12	30.67	48.24	4.08	12.01	20.14	-	-	3.58	13.81	24.44
±10°	LM [39]	12.08	11.42	35.54	70.77	80.36	5.22	15.88	26.13	3.72	2.83	19.64	51.76	71.72
	SliceMatch [32]	7.96	4.39	49.09	91.76	98.52	15.19	49.99	57.35	4.12	3.65	13.41	42.62	64.17
	CCVPE (ours)	**1.22**	**0.62**	**97.35**	**98.65**	**99.71**	**77.13**	**96.08**	**97.16**	**0.67**	**0.54**	**77.39**	**99.47**	**99.95**
360°	LM [39]	15.51	15.97	5.17	15.13	25.44	4.66	15.00	25.39	89.91	90.75	0.61	1.88	2.89
	SliceMatch [32]	9.39	5.41	39.73	**80.56**	**87.92**	13.63	40.75	49.22	**8.71**	**4.42**	**11.35**	**36.23**	**55.82**
	CCVPE (ours)	**6.88**	**3.47**	**53.30**	77.63	85.13	**25.84**	**55.05**	**68.49**	15.01	6.12	8.96	26.48	42.75

Cross-Area		↓ Localization (m)		↑ Lateral (%)			↑ Longitudinal (%)			↓ Orientation (°)		↑ Orientation (%)		
		mean	median	1m	3m	5m	1m	3m	5m	mean	median	1°	3°	5°
retrieval	CVM-Net [16]	-	-	6.96	21.55	35.24	3.58	10.45	17.53	-	-	-	-	-
	CVFT [19]	-	-	7.20	22.05	36.21	3.63	11.11	18.46	-	-	-	-	-
	SAFA [15]	-	-	9.15	27.83	44.27	4.22	11.93	19.65	-	-	-	-	-
	Polar-SAFA [15]	-	-	10.02	29.09	46.19	3.82	11.87	19.84	-	-	-	-	-
	DSM [47]	-	-	10.77	31.37	48.24	3.87	11.73	19.50	-	-	3.53	14.09	23.95
±10°	LM [39]	12.58	12.11	27.82	59.79	72.89	5.75	16.36	26.48	3.95	3.03	18.42	49.72	71.00
	SliceMatch [32]	13.50	9.77	32.43	78.98	86.44	8.30	24.48	35.57	4.20	6.61	46.82	46.82	46.82
	CCVPE (ours)	**9.16**	**3.33**	**44.06**	**81.72**	**90.23**	**23.08**	**52.85**	**64.31**	**1.55**	**0.84**	**57.72**	**92.34**	**96.19**
360°	LM [39]	15.50	16.02	5.60	16.02	25.60	5.64	15.86	25.76	89.84	89.85	0.60	1.60	2.65
	SliceMatch [32]	14.85	11.85	**24.00**	**62.52**	**72.89**	7.17	26.11	33.12	**23.64**	**7.96**	**31.69**	**31.69**	**31.69**
	CCVPE (ours)	**13.94**	**10.98**	23.42	49.15	60.46	**11.81**	**29.85**	**42.12**	77.84	63.84	3.14	9.24	14.56

Best in bold. We report mean and median localization and orientation error, as well as the percentage of test samples that have lateral/longitudinal localization or orientation error less than a threshold. The evaluation of image retrieval methods (noted with 'retrieval') is collected from [39]. In the leftmost column, ±10° denotes training and testing with an orientation prior with noise in the range [−10°; 10°], and 360° means no orientation prior by using noise from the 360° circular domain.

the ground camera, and thus it is not included in the comparison. SliceMatch and our method address orientation prediction differently. SliceMatch divides the 360° orientation space into 64 bins and selects the most probable one based on descriptors matching, while our method creates $R = 20$ orientation scores and regress the true orientation after the grid-based matching. Quantitatively, our method has a lower mean orientation error but a slightly higher median orientation error than SliceMatch. We demonstrate in our Supplementary Video that our method can smoothly track the change in the orientation of the ground camera. Grid-based solutions, such as SliceMatch, would need to densify their grid, resulting in more memory and computation needs.

Pose estimation on KITTI Same-Area: As shown in Table II Same-Area, camera pose estimation methods, LM [28], Slice-Match [32], and ours, outperform image retrieval-based methods in terms of percentages of test samples with lateral and longitudinal errors within the given thresholds. When a ±10° orientation prior is considered in both training and testing, as

assumed in [28], our method has a lower mean/median error for both localization (1.22 m / 0.62 m) and orientation estimation (0.67°/0.54°) than LM and SliceMatch.

LM needs an orientation prior to guarantee there is an overlap in the scene between its projected aerial view and the ground view for iterative optimization. As a result, LM does not work when such an orientation prior is absent, see Table II. Under this more challenging setting, the performance of both SliceMatch and our method degenerates. Our model still surpasses Slice-Match in localization performance but our model has higher errors in orientation estimation.

Qualitative results: Compared to single-mode estimators, e.g. regression-based CVR [22] and iterative optimization-based LM [28], our model shows its advantage especially when the scene contains a symmetric layout. As shown in the first two samples in Fig. 5, when there are multiple visually similar locations, e.g. zebra crossings or junctions, CVR regresses to a location between them. The iterative method LM sometimes converges to a wrong local optimum, e.g. another road, see
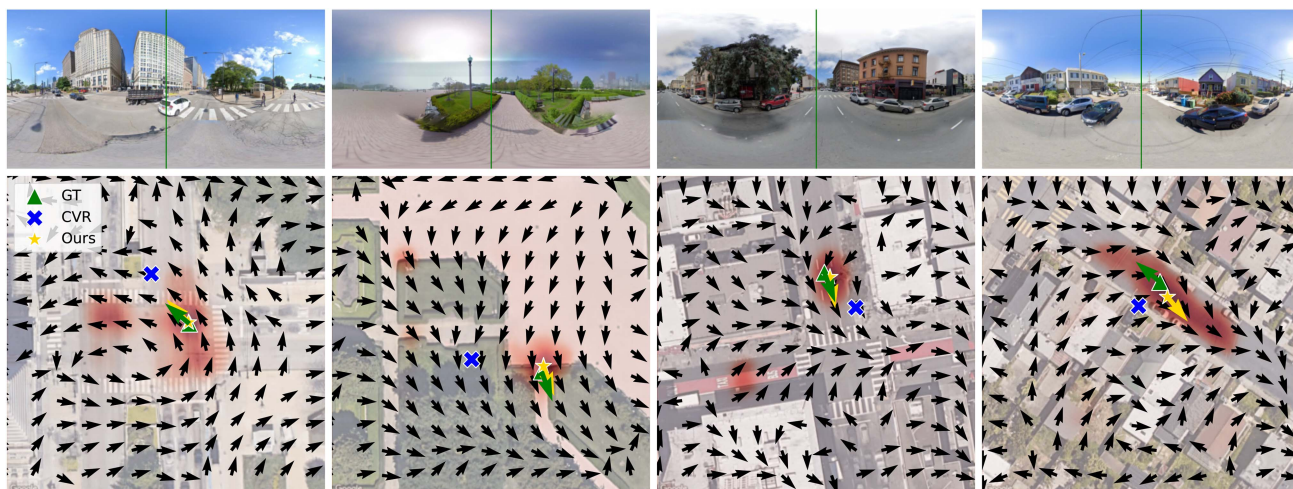
Fig. 5. Qualitative results on VIGOR. First two samples are from the Same-Area test set, and the last two samples are from the Cross-Area test set. The first three samples are success cases, the fourth shows a failure case. CVR (blue cross) receives orientation-aligned ground and aerial images and does not estimate the orientation. CCVPE (ours) selects the orientation (yellow arrow) from the prediction location (yellow star) and dense orientation map Y (black arrows). The red color shows the localization probability distribution, and the darker the color the higher the probability. The center of the ground image is always the forward direction (green vertical line), which aligns with the true orientation in the aerial view (green arrow).
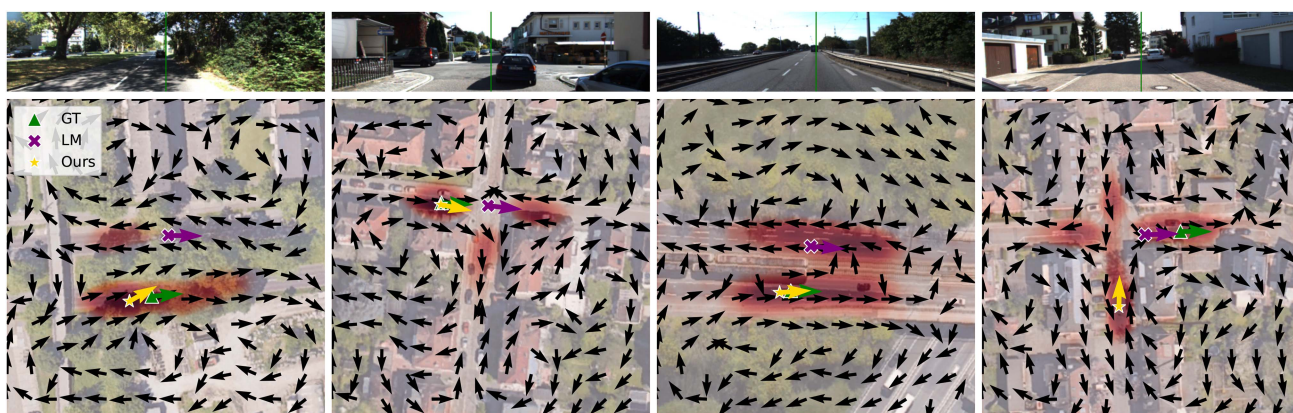

Fig. 6. Qualitative results on KITTI. First two samples are from the Same-Area test set, and the last two samples are from the Cross-Area test set. The first three samples are success cases, the fourth one shows a failure case. We provide an orientation prior with $\pm 10°$ noise to LM model, and our model does not use an orientation prior.

Fig. 6. Our model expresses its uncertainty with its multi-modal distribution to capture all probable modes, usually identifying the correct location in all datasets.

Benefiting from our joint consideration of localization and orientation, when there are multiple probable locations in our prediction, our model predicts for each of these locations the most likely orientation. As shown in the first image pair in Fig. 5, the orientation of the camera (the green line in the ground image) roughly points to the end of a zebra crossing. Our prediction suggests multiple locations, and each location predicts an orientation pointing to an end of a different zebra crossing.

At locations with a low localization probability, the ground-aerial descriptor matching has low similarity scores in all orientation channels. In this case, the orientation prediction is influenced less by the descriptor matching score but appears to follow a learned prior from the aerial view. Importantly, we

show in our ablation study that explicitly providing the descriptor matching scores is still key to good orientation prediction. Besides, we also visualize the learned features of our model using Grad-CAM [69] in our Supplementary Material.

Probabilistic prediction: Next, we evaluate the probability estimation of the baselines and our model on the VIGOR dataset. CVR [22] and Eff-CVR regress to a single location without any probability estimation. We fit a zero-mean Gaussian distribution on their predicted errors. The standard deviation of this Gauss is calculated based on their localization error on the validation set.

As shown in Table I, our model has considerably higher mean and median probability at the ground truth location than CVR [22], Eff-CVR, and SliceMatch [32]. During training, SliceMatch is optimized for discriminative descriptors, while our model is directly optimized for high probability at the ground truth location by our cross-entropy loss. In general, our model
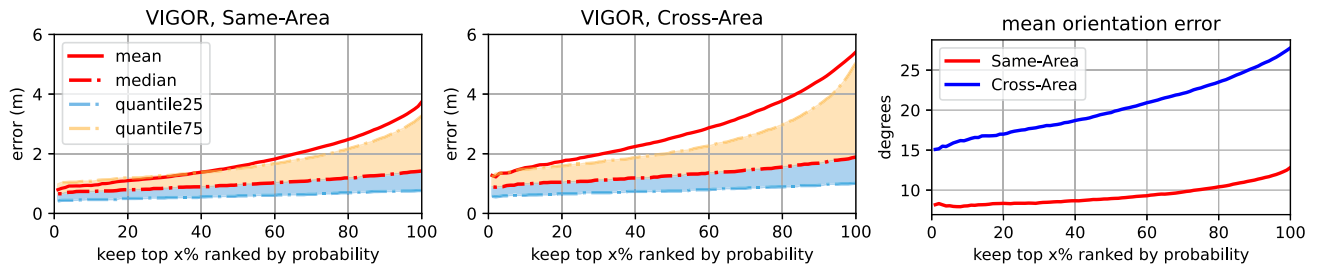
Fig. 7. Ranking CCVPE's predictions based on their estimated probabilities (tested with unknown orientation). The more confident the prediction, the lower the localization and orientation error.

is less likely to miss the ground truth location, which is an important aspect when the outputs are temporal filtered or fused with other sensor measurements.

Importantly, our probabilistic output can be used to identify predictions that potentially have large localization and orientation errors. Because we construct orientation-aware descriptors, the better an aerial descriptor matches the ground descriptor, the more likely both the location and orientation of that aerial descriptor are correct. Therefore, the localization probability can be used to filter the orientation error as well. As shown in Fig. 7, when we rank the predictions based on their predicted probabilities, the more confident predictions have in general lower localization and orientation errors. This property is important in safety-crucial applications such as autonomous driving.

F. Generalization to New Measurements Across Areas

Here we consider use cases that target operation in areas that were not covered specifically by the training data, such as driving in different cities or suburban areas.

Overall, we see a similar trend in model comparison in the Cross-Area setting as in the Same-Area setting. On VIGOR Cross-Area test set, see Table I, our model surpasses the previous SOTA SliceMatch [32] in localization by a large margin. When orientation is unknown, our median error is 66% lower than that of SliceMatch. However, our orientation error is higher than that of SliceMatch. On KITTI Cross-Area test set, see Table II, when an orientation prior with $\pm 10°$ noise presents during training and testing, our model surpasses both LM [28] and SliceMatch [32] in both localization and orientation. Without this prior, our model has lower mean and median localization error than SliceMatch, but our orientation error is higher.

Unsurprisingly, compared to the performance on the Same-Area test set, there is a performance degradation for all models. Our model could learn priors from the scene layout in the aerial image to guide its predictions. It becomes more challenging when the test aerial images are unseen. Since our predicted orientation is selected at the predicted location, the orientation error is also likely to be large when localization is wrong, see Fig. 6 last sample. We also observe there are more samples that have predicted orientation in the opposite direction on the Cross-Area test set than the Same-Area test set on both VIGOR and KITTI datasets. See the last sample of Fig. 5 for an example. In practice, when there is a prior in orientation, e.g. identified
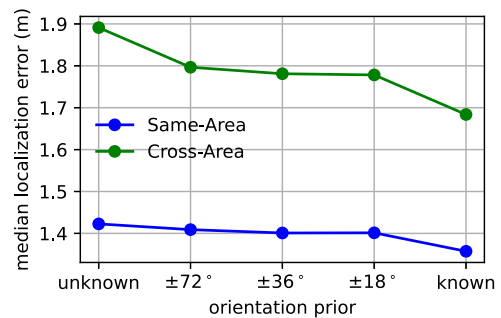

Fig. 8. Localization with different orientation priors on VIGOR Same-Area and Cross-Area test set.

by the driving direction, our model can make use of the prior to improve its prediction without retraining. We will demonstrate this in the next sub-section.

G. Effects on Orientation Prior and Image's FoV

Next, we study our model's behavior on both VIGOR Same-Area and Cross-Area test sets for inference with an orientation prior and ground images with different horizontal FoV.

Inference with an orientation prior: As described in Section III-D, our model can make use of an orientation prior without retraining. Fig. 8 shows that when a more accurate orientation prior is present, the localization performance increases accordingly. When there are multiple locations in the aerial image that match the ground image with different orientations, for example, at a crossroad, providing such an orientation prior effectively reduces the wrong matchings in our LMU and OMU modules, see examples in Fig. 9.

Inference on images with different FoVs: Moreover, our model can infer on test ground images with other horizontal FoVs than in the training data. In Fig. 10, we show the results of our models trained with ground images with various horizontal FoVs being tested on ground images with different horizontal FoVs. In general, when the FoV of the ground image increases, the information contained in the image also increases. As a result, we see a monotonic decrease in localization error when the test FoV increases for all models in Fig. 10.

An example of the predictions from our model trained with panoramic images is shown in Fig. 11. When the FoV of the test
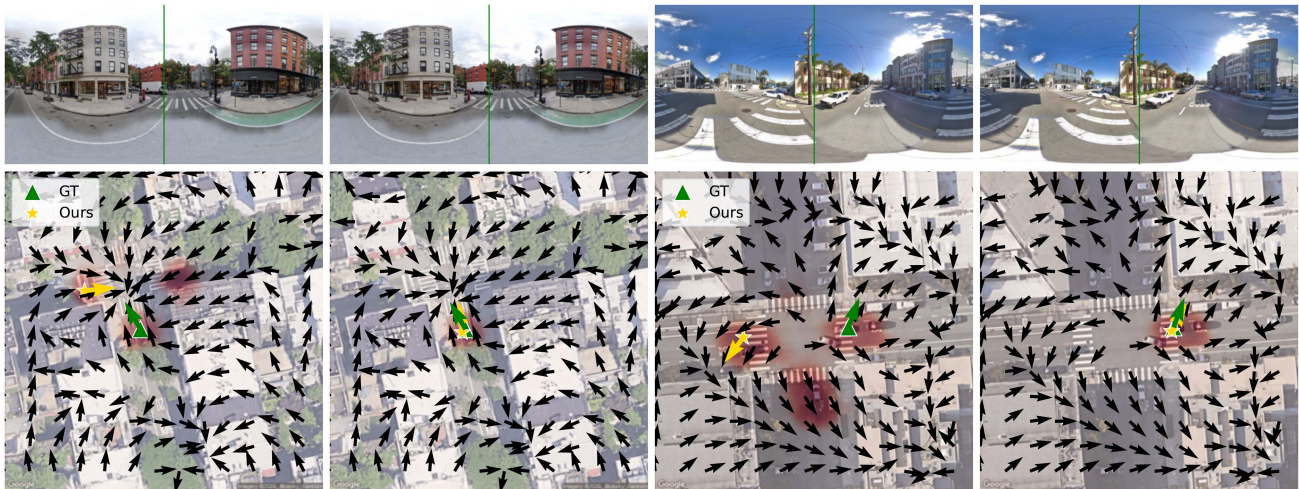
Fig. 9. Orientation prior improves our localization performance on VIGOR, Same-Area (first two image pairs) and Cross-Area (last two image pairs). The first and third image: inference without an orientation prior. The second and fourth image: inference with an orientation prior containing noise between $-36°$ and $36°$. With the prior, locations that expect a different orientation become improbable.
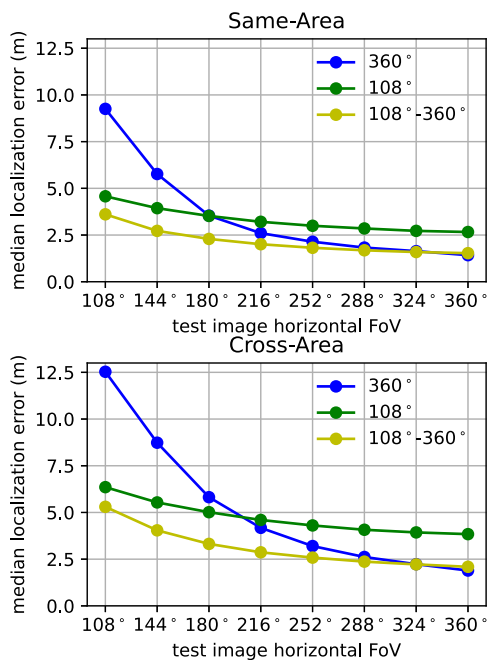

Fig. 10. Localization on images with varying horizontal FoVs on VIGOR same-area and cross-area test sets. Green/blue/yellow curves represent our model trained with horizontal FoV of $108°/360°$/between $108°$ and $360°$.

ground image is $108°$ or $180°$, our model cannot distinguish different roads based on the limited content captured by the ground image, and thus predicts a multi-modal distribution to capture the probable locations. However, the peak of the distribution is in the wrong mode and consequently, the selected orientation is also wrong. When the test FoV increases to $252°$, the peak of the output distribution is close to the ground truth location. Further increasing the FoV reduces the localization uncertainty and improves the localization. Notably, our model can always access the full scene layout information from the aerial view no

matter what the FoV of the ground view is. This example shows that the learned prior from the BEV layout solely is not enough for pose estimation, and our ground-aerial descriptor matching is crucial.

Because of the domain shift, the model trained with panoramas performs worse on images with small FoVs, compared to the model trained with images with a small FoV, see Fig. 10. Besides, we also see a steeper decrease in localization performance when the test FoV reduces for the model trained with panoramas than the model trained with images of a horizontal FoV of $108°$. Training with images with a large FoV allows the model to use features that span widely in the ground image. When those features are absent, e.g. testing with a small FoV, the performance degenerates. On the other hand, if the training images only have a small FoV, the model would not learn to use features that span wider than the FoV of the training images. Consequently, increasing the test FoV brings less benefit. To tackle this trade-off, one can train the model with images whose FoVs are randomly sampled by cropping the panorama, e.g. sampled from $108°, \ldots, 360°$. Consequently, the resulting model performs well for all tested FoVs. Interestingly, this model also has slightly better localization performance than the model trained with images with FoV of $108°$ when inference on images with FoV of $108°$. Note that this model is not used in our earlier comparison to other baselines for fairness since the baselines cannot include a similar data augmentation.

H. Ego-Vehicle Pose Estimation Across Time

On the Oxford RobotCar dataset, we deploy our model to follow the ego-vehicle over a sequence of ground-level images taken by the vehicle-mounted camera. To process a pair of input ground and aerial images on Oxford RobotCar, CCVPE takes 0.07 seconds, i.e. 14 FPS. We assume there is a rough GNSS prior that identifies which aerial patch contains the location of the ground-level image. As shown in Fig. 12, on all three test
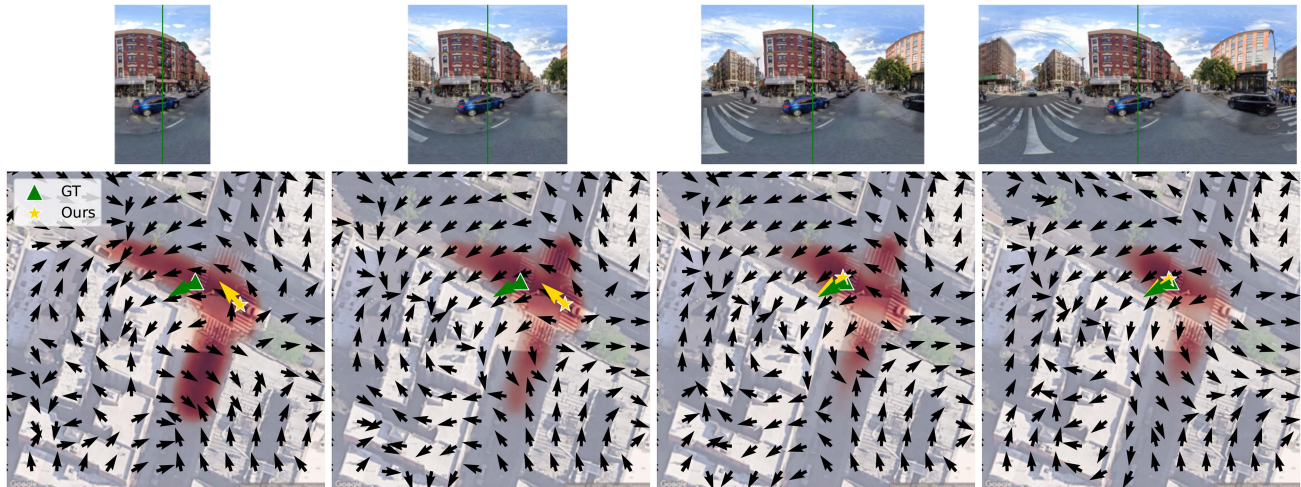
Fig. 11. Our model trained with panoramic image and inference on ground images with different FoVs (from left to right: 108°, 180°, 252°, 360°) from VIGOR samearea test set. Our dense orientation prediction is shown by black arrows. The final predicted orientation of the ground image (green vertical line) is shown by the yellow arrow.
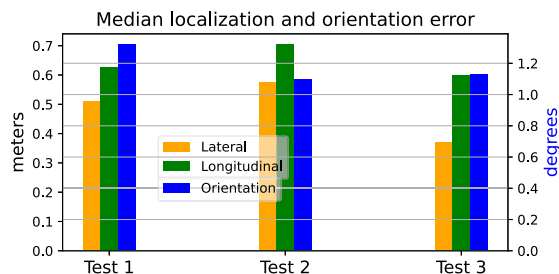

Fig. 12. Median lateral and longitudinal localization error and median orientation error on Oxford RobotCar Test 1, 2, and 3 traversals.

Fig. 13. Localization and orientation estimation on two frames in a sequence on Oxford RobotCar Test 1 traversal. Right frame: ~13 seconds after the left frame. Because the two ground-level images capture a different scene, the output orientation field in the same region A (or B) in the identical aerial images is different.

traversals, our model achieves median lateral and longitudinal localization error below 1 m and median orientation error around 1°. Notably, even though the ground images in the Oxford RobotCar dataset have a small horizontal FoV compared to the panoramas in the VIGOR dataset, our model generalizes better to new ground images along the same route across time on the former dataset than to new panoramas on the latter dataset. On the Oxford RobotCar dataset, the aerial view can provide a strong prior as the vehicle always drives along the same route and the test area is seen during training. In contrast, the panoramas in the VIGOR dataset are not always captured on the road, plus its scenes are more diverse because of its broad coverage.

On the Oxford RobotCar sequences, we can observe how the predicted orientation map adapts to ground images at different nearby locations within the same aerial view, as seen in Fig. 13. While the model learned a prior from the aerial view on the driving direction of the roads, the orientation predictions do respond to the ground image content at the high-probability locations. E.g. when the vehicle is in area A, the local orientation field points towards the junction seen in the ground view. The orientations in area A reflect a different orientation (a prior) once the vehicle moved on to area B. We provide a Supplementary Video of ego-vehicle pose estimation across time on this dataset.

I. Ablation Study

Next, we present an ablation study on the VIGOR Same-Area validation set.

Number of LMU modules: As mentioned in Section IV-D, our model has $K = 6$ LMU modules for coarse-to-fine descriptor matching for localization. Here, we study the effect of LMU modules on localization performance by removing them at low or high levels. When removing an LMU module, we modify the corresponding convolutional layer in the Localization Decoder such that it directly processes the aerial feature without any matching scores.

TABLE III
EFFECT OF LMU MODULES ON MEAN LOCALIZATION ERROR ON VIGOR SAME-AREA VALIDATION SET

VIGOR Same-Area, validation set										
K =	1	1,2	1,2,3	1,2,3,4	1,2,3,4,5	6	5,6	4,5,6	3,4,5,6	2,3,4,5,6	1,2,3,4,5,6
median error (m)	2.58	1.89	1.58	1.45	1.44	2.68	1.99	1.58	1.47	**1.42**	**1.42**

Best in bold.

(a) ground image   (b) $k=1$ (c) $k=2$ (d) $k=3$

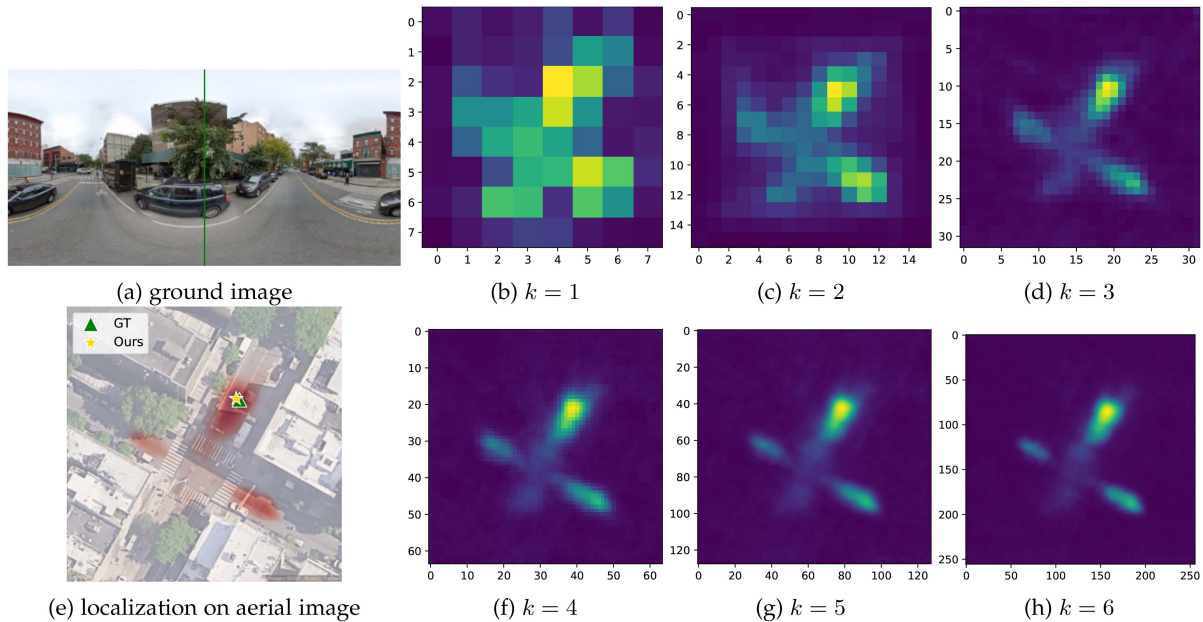(e) localization on aerial image   (f) $k=4$ (g) $k=5$ (h) $k=6$

Fig. 14.   Visualization of the max matching score map in LMU at levels 1 to 6.

As shown in Table III, the model with LMU modules at all 6 levels outperforms other variants in terms of the median localization error. When excluding the LMU modules at low or high levels, we see a consistent decrease in localization performance. Importantly, using LMU modules only at high levels, e.g. $K=6$, does not provide equally good localization performance as the models that also have LMU modules at lower levels. Directly contrasting aerial descriptors at a fine resolution is a difficult learning task. Using LMU modules at lower levels can provide a better starting point for descriptor matching at higher levels, leading to better localization performance.

Qualitatively, we see in Fig. 14 that the max matching score map inside the LMU module becomes sharper when the level k increases, but does not improve noticeably anymore after level 4, i.e. resolution 64×64, on the selected example. Quantitatively, the increase in localization performance is also less when k increases. For our main experiments, we include LMU modules at all levels, i.e. $K=6$. This setting also aligns with the commonly used coarse-to-fine formulation in other computer vision tasks [57], [58].

Other architectural variations: We first compare our model to our conference work [29]. Then, we study the effect of the backbone, the OMU module, the number of orientation channels R, and the L2-normalization before feature concatenation in the LMU and OMU modules. Finally, we test replacing the Rolling & Matching with a simple concatenation of ground and aerial features, as well as removing the contrastive learning loss $\mathcal{L}_{\mathcal{M}}$.

In our conference work [29], VGG [68] and SAFA modules [15] are used as feature extractors and feature projectors, and the ground-to-aerial matching is only conducted at the bottleneck. Thus, for a fair comparison, we include a CCVPE model with VGG [68] backbone and use the LMU module only at the bottleneck, i.e. '$K=1$, VGG'. In [29], pose estimation is achieved by comparing the model's probability estimations on differently rotated samples, while we construct orientation-aware descriptors and estimate the pose in a single forward pass. Validation results of models with different settings are summarized in Table IV. Our model with VGG [68] backbone and $K=1$ beats the approach in [29] in both localization and orientation accuracy.

The comparison between the row '$K=1$, VGG' and the row '$K=1$' in Table IV, shows that using EfficientNet-B0 [64] as the default backbone improves both localization and orientation estimation. Interestingly, we find that if we do not normalize the aerial descriptor before the feature concatenation in the LMU and OMU modules, both localization and orientation estimation performance decrease significantly. In particular, the orientation estimation becomes no better than a random guess. The magnitude of cosine similarity matching score $\mathcal{M}_k$ is between -1 and 1. Normalizing the aerial descriptors makes the magnitude of their elements stay in a similar range as $\mathcal{M}_k$. If we do not normalize the concatenated aerial descriptors, the model might not effectively use the information in matching score M_k and treat M_k as noise.

TABLE IV
CCVPE ARCHITECTURE COMPARISONS ON VIGOR SAME-AREA VALIDATION
SET

VIGOR, val.	↓ Localization (m)		↓ Orientation (°)	
	mean	median	mean	median
[29]	9.76	6.15	55.91	15.66
K=1, VGG	7.06	3.90	18.12	7.91
K=1, No norm	5.44	2.95	89.72	89.17
K=1, No infoNCE	9.37	6.04	90.59	90.53
K=1	4.93	2.61	14.68	7.50
Concat@1	13.57	11.87	89.29	89.17
K=6, No OMU	3.73	**1.40**	89.92	89.86
K=6, No infoNCE	9.11	5.67	90.22	90.17
K=6, R=2	5.35	2.40	36.74	25.39
K=6, R=4	4.51	1.80	16.06	8.57
K=6, R=5	4.21	1.74	15.11	8.42
K=6, R=10	3.85	1.51	**13.06**	6.88
K=6, R=20 (default)	**3.63**	1.42	13.11	6.61
K=6, 6 OMUs	3.78	1.43	15.34	**3.81**
Concat@6	9.66	6.63	89.60	89.03

Best in bold. 'No norm' means we do not normalize the aerial descriptors before feature
concatenation in the LMU and OMU modules. '6 OMUs' means we include the OMU
module at all levels. 'Concat' denotes direct concatenating of the ground and aerial
features instead of conducting Rolling & Matching.

Similar to concatenating aerial descriptors without normalization, excluding the OMU module and only processing the aerial descriptors also makes the orientation prediction fail. Since we randomly change the ground images' orientation during training, there is no useful prior on the orientation when only considering the aerial image. Next, we study the effect of different numbers of orientation channels R. When increasing the granularity in the orientation space, i.e. using a larger R when rolling aerial descriptors in the LMU and OMU modules, both localization and orientation estimation performance increases. Constructing aerial descriptors for more global orientation intervals not only provides more fine-grained orientation matching scores but also improves the orientation-aware features for localization. Limited by the width W' of encoded ground feature $g_e(G)$, the maximum R we can use is 20 on the VIGOR dataset. We observe a small increase in mean orientation error when increasing R from 10 to 20. Overall, $R = 20$ provides the best localization result, and therefore we used it in our main experiments (we used $R = 16$ on KITTI because the input image has a different resolution). Similar to the Localization Decoder, we tested including 6 OMU modules for all 6 levels in the Orientation Decoder. Although this setting reduces the median orientation error, we observe an increase in the localization error and mean orientation error. LMU at higher levels has finer spatial resolutions, while the granularity in orientation space is fixed, e.g. $R = 20$, in all OMU modules. Thus, we do not expect the same benefit here as in the Localization Decoder, and we use the OMU module only at the first level.

When replacing the proposed Rolling & Matching by straightforward ground-aerial feature concatenation, there is a large drop in both localization and orientation estimation performance, see the comparison between '$K = 1$' and 'Concat@1' and the comparison between '$K = 6, R = 20$' and 'Concat@6' in Table IV. When directly concatenating the ground feature with the aerial feature, the model has the additional challenge of learning that different rotated versions of the same panoramic

TABLE V
EVALUATION OF CCVPE WITH DIFFERENT LOCALIZATION LOSSES ON VIGOR
SAME-AREA VALIDATION SET

VIGOR, val.	↓ Localization (m)		↑ P@GT (10^{-3})	
	mean	median	mean	median
Cross-entropy	**3.63**	1.42	1.48	**1.00**
Wasserstein	3.75	**1.41**	**3.97**	0.00

Best in bold.

image should be located at the same place. In contrast, our Rolling & Matching design injects inductive biases into the model by using the translational equivariant ground encoder, and by forcing corresponding ground and aerial descriptors to be similar. Specifically, we ensure this rotational equivariance is kept by the OMU for orientation estimation. Therefore, when inputting different rotated versions of the same panorama, the same matching score pattern would re-occur in different orientation channels. Our Orientation Decoder still needs to learn how different permutations of matching scores translate to the orientation vector field, but this matching volume has a relatively low number of channels compared to concatenated ground and aerial features. The LMU's inductive bias is to be *invariant* to different ground camera's orientations, which is achieved by taking the maximum over orientation channels. None of these orientation and localization-specific inductive biases are present in the concatenation approach, which explains the large difference in performance.

Importantly, our Rolling & Matching is empowered by the orientation-aware ground and aerial descriptors. If we do not enforce the orientation awareness for the aerial descriptor by removing the infoNCE losses, both localization and orientation prediction performance of the model decreases significantly, see rows with 'No infoNCE' in Table IV.

Loss on localization heat map: Using our best model architecture, we compare the cross-entropy loss and Wasserstein distance-based loss for localization and uncertainty estimation. In Table V, we show the mean and median localization error and the predicted probability at the ground truth pixel of models trained with different losses.

The model trained with cross-entropy loss has a lower mean localization error than the model trained with Wasserstein distance-based loss. Notably, the model trained with Wasserstein distance-based loss outputs localization distributions that are very sharp. Biased by a few accurate predictions, the mean probability at the ground truth pixel of this model is higher than that of the model trained with cross-entropy. However, the median probability at the ground truth pixel is near zero, indicating many of the ground truth locations receive little probability mass. In temporal filtering or multi-sensor fusion, fusing such predictions might make the system miss the ground truth location. Besides, we also observe that training with Wasserstein distance-based loss makes the output distribution less indicative of localization and orientation errors. This reduces its practicality in safety crucial applications where the outliers in prediction should be filtered out. Thus, we used the cross-entropy loss as our localization loss $\mathcal{L}_D$.

J. Runtime Analysis

First, we study how the proposed Rolling & Matching influences the runtime of our method. On the VIGOR dataset, when increasing the number of orientation bins ($R = 2, 4, 5, 10, 20$) for Rolling & Matching in all LMU and OMU modules, the inference speed of our method decreases slightly (18, 17, 17, 17, 15 FPS). Since the Rolling & Matching is a convolution process between ground and aerial descriptors, it can be done efficiently.

Next, we compare the runtime of our method to that of previous state-of-the-art methods. To include more baselines, the comparison is done on the KITTI dataset. On the same device (a single V100 GPU), our method takes 0.042 s to process a pair of input images (24 FPS) on the KITTI dataset, which is slower than SliceMatch's 156 FPS [32] but faster than LM's 0.59 FPS [28]. Importantly, even though SliceMatch runs faster, CCVPE is considerably more accurate in localization. Note that the authors of [33] evaluated the runtime of their method on the KITTI-360 dataset with a more advanced GPU (RTX6000), and their method runs at approximately 2–3 Hz [33], which is slower than CCVPE.

V. CONCLUSION

In this paper, the novel Convolutional Cross-View Pose Estimation method (CCVPE) was proposed. CCVPE exploits the strength of a translational equivariant feature encoder and of contrastive learning to learn orientation-aware descriptors for joint localization and orientation estimation. Instead of estimating a single location, its Localization Decoder outputs a multi-modal distribution to capture the underlying localization uncertainty. The Localization Matching Upsampling (LMU) and Orientation Matching Upsampling (OMU) modules were devised to summarize orientation invariant localization cues and orientation-dependent information from the descriptor matching result when upsampling the aerial feature maps inside two separate decoders. The Orientation Decoder outputs a dense orientation vector field that is conditioned on the localization distribution. Thus, CCVPE's orientation prediction becomes multi-modal when there are multiple modes in the localization distribution.

CCVPE achieves 72% and 36% lower median localization errors (1.42 m and 3.47 m) than the previous SOTA (5.07 m and 5.41 m) on the VIGOR and KITTI datasets, and it has comparable orientation estimation accuracy. Importantly, CCVPE can work with ground images with different horizontal FoVs and incorporate an orientation prior to improve the localization without re-training. Its probabilistic output can be used to filter out predictions that potentially have large localization and orientation errors, yielding better practicality than the baselines that do not have a probability estimate. We demonstrated on traversals collected at different times in the Oxford RobotCar dataset that CCVPE can estimate the pose of ego-vehicle at 14 FPS with a median lateral and longitudinal error below 1 m and a median orientation error around $1°$, bringing cross-view pose estimation methods closer to the requirement of autonomous driving of < 0.3 m lateral and longitudinal localization accuracy. Future work will address applying temporal filtering on the single frame estimates and multi-sensor fusion to further increase the pose estimation accuracy. Besides, more efficient descriptor construction and matching strategies will be studied.

REFERENCES

[1] S. Thrun, "Probabilistic robotics," *Commun. ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[2] B. Ben-Moshe et al., "Improving accuracy of GNSS devices in urban canyons," in *Proc. 23rd Annu. Can. Conf. Comput. Geometry*, 2011, pp. 511–515.

[3] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, "A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence," 2020, *arXiv: 2006.12567.*

[4] S. Lowry et al., "Visual place recognition: A survey," *IEEE Trans. Robot.*, vol. 32, no. 1, pp. 1–19, Feb. 2016.

[5] J. Janai et al., "Computer vision for autonomous vehicles: Problems, datasets and state of the art," *Found. Trends Comput. Graph. Vis.*, vol. 12, no. 1/3, pp. 1–308, 2020.

[6] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12716–12725.

[7] I. A. Barsan, S. Wang, A. Pokrovsky, and R. Urtasun, "Learning to localize using a LiDAR intensity map," in *Proc. Annu. Conf. Robot Learn.*, 2018, pp. 605–616.

[8] X. Wei, I. A. Bârsan, S. Wang, J. Martinez, and R. Urtasun, "Learning to localize through compressed binary maps," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10316–10324.

[9] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-Net: Towards learning based LiDAR localization for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6389–6398.

[10] H. Wang, C. Xue, Y. Zhou, F. Wen, and H. Zhang, "Visual semantic localization based on HD map for autonomous vehicles in urban scenarios," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 11255–11261.

[11] C. Guo, M. Lin, H. Guo, P. Liang, and E. Cheng, "Coarse-to-fine semantic localization with HD map for autonomous driving in structural scenes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 1146–1153.

[12] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5007–5015.

[13] S. Workman, R. Souvenir, and N. Jacobs, "Wide-area image geolocalization with aerial reference imagery," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 3961–3969.

[14] S. Workman and N. Jacobs, "On the location dependence of convolutional neural network features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2015, pp. 70–78.

[15] Y. Shi, L. Liu, X. Yu, and H. Li, "Spatial-aware feature aggregation for image based cross-view geo-localization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 10090–10100.

[16] S. Hu, M. Feng, R. M. Nguyen, and G. Hee Lee, "CVM-net: Cross-view matching network for image-based ground-to-aerial geo-localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7258–7267.

[17] K. Regmi and M. Shah, "Bridging the domain gap for ground-to-aerial image matching," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 470–479.

[18] S. Cai, Y. Guo, S. Khan, J. Hu, and G. Wen, "Ground-to-aerial image geo-localization with a hard exemplar reweighting triplet loss," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 8391–8400.

[19] Y. Shi et al., "Optimal feature transport for cross-view image geo-localization," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11990–11997.

[20] S. Zhu, T. Yang, and C. Chen, "Revisiting street-to-aerial view image geo-localization and orientation estimation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2021, pp. 756–765.

[21] A. Toker, Q. Zhou, M. Maximov, and L. Leal-Taixe, "Coming down to earth: Satellite-to-street view synthesis for geo-localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 6488–6497.

[22] S. Zhu, T. Yang, and C. Chen, "VIGOR: Cross-view image geo-localization beyond one-to-one retrieval," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3640–3649.

[23] H. Yang, X. Lu, and Y. Zhu, "Cross-view geo-localization with layer-to-layer transformer," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 29009–29020.

[24] S. Zhu, M. Shah, and C. Chen, "TransGeo: Transformer is all you need for cross-view image geo-localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1162–1171.

[25] R. Rodrigues and M. Tani, "Global assists local: Effective aerial representations for field of view constrained image geo-localization," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2022, pp. 3871–3879.

[26] Z. Xia, O. Booij, M. Manfredi, and J. F. P. Kooij, "Geographically local representation learning with a spatial prior for visual localization," in *Proc. Eur. Conf. Comput. Vis. Workshops*, Springer, 2020, pp. 557–573.

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

[28] Y. Shi and H. Li, "Beyond cross-view image retrieval: Highly accurate vehicle localization using satellite image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 17010–17020.

[29] Z. Xia, O. Booij, M. Manfredi, and J. F. P. Kooij, "Visual cross-view metric localization with dense uncertainty estimates," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 90–106.

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

[31] Y. Hou, Y. Yang, J. Wang, and M. Fu, "Road extraction assisted offset regression method in cross-view image-based geo-localization," in *Proc. IEEE 25th Int. Conf. Intell. Transp. Syst.*, 2022, pp. 2934–2940.

[32] T. Lentsch, Z. Xia, H. Caesar, and J. F. P. Kooij, "SliceMatch: Geometry-guided aggregation for cross-view pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 17225–17234.

[33] F. Fervers, S. Bullinger, C. Bodensteiner, M. Arens, and R. Stiefelhagen, "Uncertainty-aware vision-based metric cross-view geolocalization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 21621–21631.

[34] T. G. Reid et al., "Localization requirements for autonomous vehicles," *SAE Int. J. Connected Automated Veh.*, vol. 2, no. 3, pp. 173–190, 2019.

[35] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *Int. J. Roboti. Res.*, vol. 36, no. 1, pp. 3–15, 2017.

[36] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.

[37] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[38] L. Liu and H. Li, "Lending orientation to neural networks for cross-view geo-localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5624–5633.

[39] Y. Shi, X. Yu, L. Liu, D. Campbell, P. Koniusz, and H. Li, "Accurate 3-DoF camera geo-localization via ground-to-satellite image matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 2682–2697, Mar. 2023.

[40] S. Li, Z. Tu, Y. Chen, and T. Yu, "Multi-scale attention encoder for street-to-aerial image geo-localization," *CAAI Trans. Intell. Technol.*, vol. 8, no. 1, pp. 166–176, 2022.

[41] K. Regmi and A. Borji, "Cross-view image synthesis using conditional GANs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3501–3510.

[42] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5297–5307.

[43] T. Wang et al., "Each part matters: Local patterns facilitate cross-view geo-localization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 2, pp. 867–879, Feb. 2022.

[44] Y. Shi, X. Yu, S. Wang, and H. Li, "CVLNet: Cross-view semantic correspondence learning for video-based camera localization," in *Proc. IEEE/CVF Asian Conf. Comput. Vis.*, 2022, pp. 652–669.

[45] N. N. Vo and J. Hays, "Localizing and orienting street views using overhead imagery," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2016, pp. 494–509.

[46] M. Zhai, Z. Bessinger, S. Workman, and N. Jacobs, "Predicting ground-level scene layout from aerial imagery," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 867–875.

[47] Y. Shi, X. Yu, D. Campbell, and H. Li, "Where am I looking at? Joint location and orientation estimation by cross-view matching," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4064–4072.

[48] S. Hu and G. H. Lee, "Image-based geo-localization using satellite imagery," *Int. J. Comput. Vis.*, vol. 128, no. 5, pp. 1205–1219, 2019.

[49] L. M. Downes, D.-K. Kim, T. J. Steiner, and J. P. How, "City-wide street-to-satellite image geolocalization of a mobile ground agent," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 11102–11108.

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

[51] W. Hu et al., "Beyond geo-localization: Fine-grained orientation of street-view images by cross-view matching with satellite imagery," in *Proc. ACM Multimedia*, 2022, pp. 6155–6164.

[52] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929.*.

[53] H. Howard-Jenkins, J.-R. Ruiz-Sarmiento, and V. A. Prisacariu, "LaLaLoc: Latent layout localisation in dynamic, unvisited environments," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10107–10116.

[54] H. Howard-Jenkins and V. A. Prisacariu, "LaLaLoc : Global floor plan comprehension for layout localisation in unvisited environments," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 693–709.

[55] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in MICCAI. Springer, 2015, pp. 234–241.

[56] Z. Min et al., "LASER: Latent space rendering for 2D visual localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11122–11131.

[57] A. Dosovitskiy et al., "FlowNet: Learning optical flow with convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015.

[58] N. Mayer et al., "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4040–4048.

[59] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8934–8943.

[60] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv: 1807.03748*.

[61] G. Monge, "Mémoire sur la théorie des déblais et des remblais," in *Mem. Math. Phys. Acad. Royale Sci.*. Paris, France: Imprimerie Royale, 1781, pp. 666–704.

[62] C. Frogner, C. Zhang, H. Mobahi, M. Araya, and T. A. Poggio, "Learning with a Wasserstein loss," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 2053–2061.

[63] W. Maddern et al., "Real-time kinematic ground truth for the Oxford RobotCar dataset," 2020, *arXiv: 2002.10152*.

[64] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[65] J. Deng et al., "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

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

[67] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, "Regularization of neural networks using dropconnect," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1058–1066.

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

[69] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2017, pp. 618–626.

Zimin Xia received the BSc degree in geomatics engineering from Wuhan University, in 2016, and the MSc degree in geomatics engineering from the University of Stuttgart, in 2019. Currently, he is working toward the PhD degree with the Intelligent Vehicles Group, TU Delft. His research interests include visual localization, ground-aerial image matching, representation learning, and self/semi-supervised learning.

Olaf Booij received the MSc degree in artificial intelligence on the topic of spiking neural networks, and the PhD degree on vision-based mapping from the University of Amsterdam, in 2004 and 2011, respectively. Afterwards, he worked as a researcher and team lead with Fugro Intersite, developing robust computer vision methods for offshore and sub-sea and with TomTom as deep learning expert and manager software engineering for automated map making. Currently, he's a guest researcher with the Computer Vision lab, TU Delft.

Julian F. P. Kooij (Member, IEEE) received the PhD degree from the University of Amsterdam on visual detection and path prediction for vulnerable road users, in 2015. Afterwards, he joined TU Delft, first in the Computer Vision lab and since 2016 in the Intelligent Vehicles group where he is currently an Associate Professor. His research interests include deep representation learning and probabilistic models for multi-sensor localisation, object detection, and forecasting of urban traffic.