Cross-View Matching for Vehicle Localization by Learning Geographically Local Representations

Zimin Xia[1], Olaf Booij[2], Marco Manfredi[2] and Julian F. P. Kooij[1]

***Abstract*— Cross-view matching aims to learn a shared image representation between ground-level images and satellite or aerial images at the same locations. In robotic vehicles, matching a camera image to a database of geo-referenced aerial imagery can serve as a method for self-localization. However, existing work on cross-view matching only aims at global localization, and overlooks the easily accessible rough location estimates from GNSS or temporal filtering. We argue that the availability of coarse location estimates at test time should already be considered during training. We adopt a simple but effective adaptation to the common triplet loss, resulting in an image representation that is more discriminative within the geographically local neighborhood, without any modifications to a baseline deep neural network. Experiments on the CVACT dataset confirm that the improvements generalize across spatial regions. On a new benchmark constructed from the Oxford RobotCar dataset, we also show generalization across recording days within the same region. Finally, we validate that improvements on these image-retrieval benchmarks also translate to a real-world localization task. Using a particle filter to fuse the cross-view matching scores of a vehicle's camera stream with real GPS measurements, our learned geographically local representation reduces the mean localization error by 17% compared to the standard global representation learned by the current state-of-the-art.**

***Index Terms*— Deep representation learning, cross-view image matching, vehicle localization, particle filter**

I. Introduction

With the rise of camera-equipped vehicles, visual localization has become a key research topic in autonomous driving. No matter how the map is presented, most visual localization methods explicitly or implicitly match an input image to a representation of the map. For instance, image retrieval-based localization locates the query image by matching it to the geo-referenced images in a shared representation space. An increasingly popular variant is cross-view matching-based localization [1], [2], [3], [4], [5], [6], where the query ground-view image is compared to aerial/satellite imagery. This setting enjoys the reliable representation and dense coverage of the environment from the overhead view. Plus, large databases are nowadays readily available [6], [3].

In the robotics domain, localization is traditionally addressed using specialized sensors, e.g. Global Navigation Satellite Systems (GNSS). Unfortunately, the horizontal positioning error of stand-alone GNSS can reach tens of meters [7], [8] near high rise buildings or under trees, due to the multipath effect. In practice, the GNSS localization is often fused with measurements from other sensors, e.g. wheel odometry or camera, and combined with temporal filtering.

[1]Intelligent Vehicles Group, Technical University Delft, The Netherlands, {z.xia,j.f.p.kooij}@tudelft.nl
[2]TomTom, Amsterdam, The Netherlands, {olaf.booij,marco.manfredi}@tomtom.com

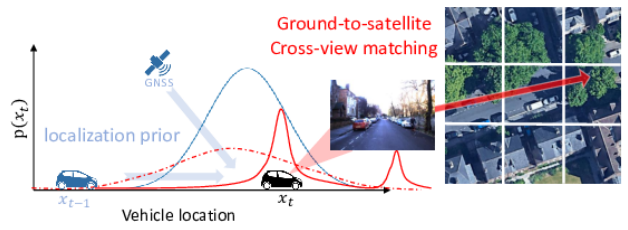


Fig. 1: Vehicles can use cross-view matching between camera images and satellite patches for self-localization, resulting in a geo-global localization estimate (red dashed curve). However, a coarse localization prior (blue curve) is often already available from other sensors or temporal integration. We exploit this prior during training to obtain a more discriminative model within the local area (red solid curve).

We observe, however, that there are substantial gaps in how the localization task is addressed in mobile robotics and autonomous driving, and the state-of-the-art image retrieval-based localization techniques.

First, image retrieval-based localization is often treated as a substitute for GNSS for global place recognition [9], [10], though in practice GNSS and temporal filtering can provide a complementary coarse location estimate [11].

Second, existing cross-view matching benchmarks [3], [12] measure how the model generalizes to new areas, as they split the data according to its geographic region. However, in practice, we *can* have satellite images of the test region available during training, especially for a navigation task with geo-localized road information, which already presupposes that the target region is known. Therefore, an equally relevant question is how the learned representation generalizes to new ground-level observations on different days in the same area.

Third, many cross-view image matching-based localization methods [4], [3], [13], [14] are evaluated solely using image retrieval metrics, such as recall@K. Such metrics do not measure the actual localization capability, and do not reflect that a ground image's view does not necessarily correspond to any satellite image's center location, or could even coincide with multiple overlapping satellite images.

To address the observed gaps, we exploit the context of cross-view matching within a localization system. Since other components, e.g. GNSS and temporal filtering, will already provide a coarse location estimate, we propose to train cross-view matching to be especially discriminative within this local region of uncertainty, rather than differentiating far-away areas that the prior would already discard, see Figure 1.

The main contributions of our work include: (i) We demonstrate that our previously proposed local triplet loss [15] can improve two state-of-the-art cross-view matching methods, and study the impact of its hyperparameters and weighting function. (ii) We augment the well-known Oxford RobotCar dataset with a map composed of satellite images to serve as a new dense cross-view localization benchmark to test generalization across recording days. We also test on data from the existing CVACT benchmark, for which we propose new splits, to test generalization across regions. On both benchmarks, we show quantitative improvements over the state-of-the-art, and provide qualitative results to show the difference between encoded geo-local and geo-global features. (iii) We test our approach in a real-world scenario where query images are matched against satellite images distributed evenly in the target area, and the cross-view matching-based localization is fused in a particle filter with priors from actual GPS measurements. We demonstrate superior localization accuracy and robustness against the baseline cross-view matching fused with GPS.

II. Related Work

Boosted by the remarkable representation power of deep learning, learned image representations [16] are gradually replacing the traditional hand-crafted feature descriptors [17], [18] in image retrieval. In visual localization, a majority of works target ground-level image matching [19], the scalability of those methods is limited by the sparse coverage of geo-tagged ground-level images.

The abundance of high-resolution aerial and satellite imagery motivates another variant, cross-view matching-based localization, which has shown potential in city-scale or even country-scale geo-localization in past years [6], [2], [5]. Commonly, this task is addressed using metric learning. CVM-Net [1] tests different feature extractors together with NetVLAD [16] to map input images into a shared representation space. The final matching score for a satellite-ground image pair is given by the distance measurement of two NetVLAD descriptors. [3] proposes to encode the azimuth and altitude of each pixel in the query image as extra feature channels to include orientation information in the matching.

One of the main challenges in cross-view matching comes from the large difference between the two views. The Spatial-Aware Feature Aggregation (SAFA) network [4] introduces a polar transformation pre-processing, that warps satellite images w.r.t. ground images to reduce the domain gap. DSM [13] adopts the same pre-processing and proposes a network architecture to jointly estimate the orientation of the ground image. Instead of explicitly warping the satellite image, the CVFT framework [14] proposes a feature transport module to convert the features from ground image towards satellite domain inside an end-to-end network. In [20], the authors seek another way to minimize the domain gap. They use a conditional GAN to generate a synthetic satellite image from the ground-level panorama and then match both to the satellite images in the database. In contrast, [21] tackles the ground-to-satellite discrepancy by learning to synthesize street views from satellite inputs. VIGOR [22] addresses the situation where the ground and satellite images are not perfectly geographically aligned, and proposes an end-to-end network to retrieve a nearby satellite image and then regress the location offsets.

Recently, cross-view matching methods have been employed in vehicle localization as a replacement of GNSS [9], [10] thanks to their decent global localization performance. In [9], the authors first tessellate the satellite map into grids at 5m intervals, and then localize the query ground image w.r.t. the grids using CVM-Net [1]. A mean position error at around 20m is achieved by combining the cross-view matching with visual odometry using a particle filter.

Still, all these approaches focus on the challenge of learning *globally* discriminative localization features, without considering in the training that, in practice, a good localization prior can be obtained from GNSS and temporal filtering.

III. Methodology

We start by reviewing the task of cross-view matching and the triplet loss used in the baseline and related work. After this, our geo-local loss is introduced. Finally, we discuss how we combine cross-view matching and GNSS measurements in a particle filter for online vehicle localization.

A. Cross-view matching task

Given a *ground-level* query image G_q, the objective of cross-view matching is to select the closest *satellite* image from the target dataset $\mathbf{S} = (S_1, S_2, \cdots)$. Each satellite image S_i here covers a fixed-sized square area of the Earth's surface, and the 2D geographic location $\pi(S_i) \in \mathbb{R}^2$ of the center of the square is known. The matching is done in a representation space, where the satellite images and query are mapped into normalized image descriptors using mapping function $f(\cdot)$ and $g(\cdot)$ respectively. The descriptor of the best-matched satellite image should have the smallest squared Euclidean distance to the descriptor of the query.

B. Baseline architecture and geo-global triplet loss

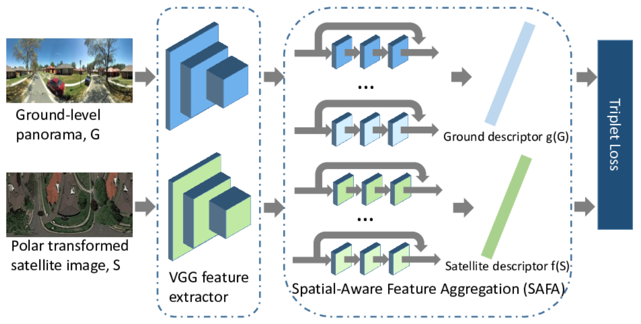

Ground-level panorama, G
Polar transformed satellite image, S
VGG feature extractor
...
Ground descriptor g(G)
Satellite descriptor f(S)
Spatial-Aware Feature Aggregation (SAFA)
Triplet Loss

Fig. 2: The architecture of our baseline cross-view matching method, SAFA [4].

While our approach is generic, we will use the state-of-the-art SAFA method [4] as our baseline. As shown in

Figure 2, the mapping functions $f(\cdot)$ and $g(\cdot)$ in SAFA are implemented as a 16-layer VGG feature extractor and 8 separate spatial-aware feature aggregation modules [4]. They map input images to 4096-dimensional descriptors. Two network branches without weight-sharing are trained on image pairs $\mathcal{X} = \{(S_1, G_1), (S_2, G_2), \cdots\}$ using a soft-margin triplet loss for two related matching objectives,

$$l_1(i,j) = \log(1 + e^{\gamma(d_{i,i} - d_{i,j})}), \quad \text{(satellite-to-ground)} \quad (1)$$

$$l_2(i,j) = \log(1 + e^{\gamma(d_{i,i} - d_{j,i})}). \quad \text{(ground-to-satellite)} \quad (2)$$

Here $d_{i,j} = ||f(S_i) - g(G_j)||_2^2$ is the squared Euclidean distance between the descriptors, and γ is a hyperparameter to adjust the gradient of the loss. The final loss is the average of $l_1(i,j)$ and $l_2(i,j)$. For a minibatch $\mathcal{B} \subseteq \mathcal{X}$ of N pairs, the loss terms can be efficiently computed by performing the forward passes $f(S_i)$ and $g(G_i)$ only once for all N samples, and then just computing N^2 squared Euclidean distances $d_{i,j}$ of all combinations i, j.

An important aspect of the baseline is that it selects minibatches from the training data by randomly shuffling *all* samples in each epoch, thus any two pairs are equally likely to co-occur in the batches, independently from their geographic proximity. This triplet loss thus learns a *globally* discriminative representation.

C. Training with a geo-local triplet loss

Vehicle localization provides at every time step a coarse localization estimate from fusing and filtering past sensor measurements. We therefore seek to exploit knowledge of a coarse prior already during training, and will consider two adaptations to the baseline loss, namely *geo-distance weighted loss terms* and *local minibatches* [15].

1) Geo-distance weighted loss terms: The triplet losses of Eq. (1) and (2) are multiplied with a weight $w_{geo}(i,j)$ that scales their contribution based on the Euclidean distance $\delta_{i,j} = ||\pi(S_i)\text{-}\pi(S_j)||_2$ (in meters) between their geographic positions $\pi(S_i)$ and $\pi(S_j)$ [15],

$$w_{geo}(i,j) = p_r(\delta_{i,j}) \cdot (1 - e^{-\delta_{i,j}{}^2/(2\sigma_{\text{geo}}^2)}). \quad (3)$$

The first term $p_r(\delta_{i,j})$ models a prior on the coarse localization error, which is assumed to be maximally r meters. Importantly, it should force training to ignore triplets with $\delta_{i,j} > r$ in favor of nearby ones. We will consider two options for p_r. Option 1 uses a *step* function to weigh all triplets 1 if $\delta_{i,j} \leq r$ and 0 otherwise [15], see the green dotted line in Figure 3. Option 2 uses instead a *Gaussian* function with std.dev. $r/3$ such that the weight smoothly drops to (nearly) zero at r meters, see red dotted line in Figure 3. The second term is added to down-weight the loss on geographically *nearby* samples to prevent the model from treating two nearly identical satellite images, e.g. with 1-meter distance, one as positive and the other one as negative. The hyperparameter σ_{geo} controls the smoothness of this weight reduction.

The full weight function $w_{geo}(i,j)$ is thus the product of both terms, and scaled such that the weight at its maximum is 1, see the green/red solid lines in Figure 3 for the final weight function with a step/Gaussian decay.

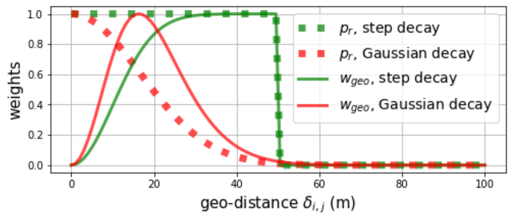


Fig. 3: The weight decay options (dashed) and resulting weight functions $w_{geo}(i,j)$ (solid), here shown as an example of $r = 50$m and $\sigma_{\text{geo}} = 10$m.

2) Local minibatches: Using the geo-distance weighted loss term, most randomly picked pairs from the training data would have zero weight as they are likely to be at distant geographic locations, especially when the mapped area is large. We therefore construct local minibatches that only contain pairs from nearby geographic locations, using the following procedure:

1) pre-compute before training for each pair $P_i = (S_i, G_i)$ the local neighborhood of pairs within a geographic radius of r meters, i.e.

$$\mathcal{N}_r(i) = \{(S_j, G_j) \mid i \neq j \wedge \delta_{i,j} \leq r\} \subset \mathcal{X}. \quad (4)$$

2) At the start of an epoch, create a fresh set $\tilde{\mathcal{X}}$ containing all training samples, $\tilde{\mathcal{X}} \leftarrow \mathcal{X}$, representing the still unused samples in this epoch.
3) To create a new minibatch $\mathcal{B}$ of size N, first randomly pick a pair P_i from pool $\tilde{\mathcal{X}}$, and then uniformly pick without replacement the remaining $N-1$ samples from the neighborhood set $\mathcal{N}_r(i)$. All picked samples are removed from the epoch's pool, $\tilde{\mathcal{X}} \leftarrow \tilde{\mathcal{X}}/\mathcal{B}$. Once $\tilde{\mathcal{X}}$ is empty, a new epoch is started.

Since all pairs j in the batch are by definition within distance r from the first sampled pair i, two samples j and j' in the minibatch can be *at most* a distance of $2r$ meters apart. This local minibatch formulation greatly increases the chance that many pairs in the minibatch are also within each other's r-meter radius, and thus largely reduces the chance of near-zero geo-distance weighted loss terms. Note that overall each pair occurs in *at most* one minibatch per epoch. Pairs without enough neighbors will not be used.

Note that r is a measure of the coarse prior's maximum uncertainty, it thus not an optimizable hyperparameter but given by the targeted localization use-case. To avoid minibatches with too few samples, the selected training data should contain at least $N-1$ neighbors within a radius of r of each sample.

D. Particle filter-based localization

We here describe how online vehicle localization could use cross-view matching at test time, and fuse it with real-world GNSS measurements in a temporal filter, as opposed to replacing the GNSS (e.g. [9], [10]). Since the underlying distribution of the localization results will be multi-modal, we capture this distribution by constructing a particle filter-based localization pipeline [23] that combines the cross-view matching and GPS positioning. We assume the availability of a satellite images $\mathbf{S}_{grid}$ which cover the target area centered around points on a dense regular grid.

Each particle m has a 4D state vector $x^{[m]}$ containing Easting, Northing, forward velocity, and yaw in the map's coordinate frame. Let χ_t denote the set of $M = 2000$ particles at step t. At $t = 0$, all particles are initialized at the GPS measured location with random yaw between -180° and 180° and a random velocity between 0 and $5\text{m}/s$. For $t > 0$, a prediction is made for each particle χ_{t-1} using a fixed velocity motion model with Gaussian acceleration and steering noise. The particles are then weighted by the measurement model, and finally resampled proportional to weight to obtain χ_t. As filter output, we take the median of each element in the state vector over all particles in χ_t.

The measurement model weighs each particle ${x_t}^{[m]}$ according to the query image $G_{q,t}$ and the raw GNSS positioning $\pi(G_{q,t})$. Assume that the GPS uncertainty follows a Gaussian distribution with known standard deviation σ_{gps}. We use $3\sigma_{\text{gps}}$ as a confidence threshold of this distribution, and select from the database the satellite image $\mathbf{S}_{local} \subseteq \mathbf{S}_{grid}$ within the threshold, i.e. a 2D circle centered at $\pi(G_{q,t})$ with a radius of $3\sigma_{\text{gps}}$ meters. Particles outside this circle are directly discarded, and only satellite images $S_j \in \mathbf{S}_{local}$ are compared to $G_{q,t}$ to compute their cross-view matching score $e^{-d_{j,q}}$. Given $\pi(m)$, the Easting and Northing location of ${x_t}^{[m]}$, let $e^{-d_{m,q}}$ be the geo-distance-based bi-linear interpolation of the matching scores for the 4 satellite images at the grid points around $\pi(m)$. The particle's weight is then,

$$ {w_t}^{[m]} = \frac{e^{-d_{m,q}}}{\sum_{j \in \mathbf{S}_{local}} e^{-d_{j,q}}} \cdot e^{-||\pi(m) - \pi(G_{q,t})||_2^2 / (2{\sigma_{\text{gps}}}^2)}. \quad (5) $$

Here the first term computes the probability of the query being located at $\pi(m)$ as given by cross-view matching. This probability equals the matching score at $\pi(m)$ over the sum of scores between the query and all satellite images in $\mathbf{S}_{local}$. The second term in the equation measures the likelihood of $\pi(m)$ being correct location according to the raw GPS measurement. Eq. (5) thus presents a straightforward sensor fusion of the visual cross-view matching and GNSS localization measurements, and is applicable irrespective if the matching network is trained geo-local or geo-global.

Unfortunately, GPS measurements inevitably carry huge errors in extreme cases, for example when no satellites are in sight. Motivated by the outlier rejection found in [24], [23], we handle such situations by not using the GPS measured location at step t when this is over $r_{outlier}$ meters apart from the GPS measured location at step $t-1$ or if there is no valid GPS measurement at this step. Instead of the raw GPS, we then use the estimated location at step $t-1$ as $\pi(G_{q,t})$. We set the $r_{outlier}$ to $3 \cdot \sigma_{\text{gps}} + v_{t-1} \cdot \Delta t$, where v_{t-1} is the estimated velocity at previous step and Δt is the time interval.

IV. Experiments

We compare our geo-local representation learning to the standard geo-global representation learning [4] in two scenarios, namely, generalization across regions and generalization across time. Besides quantitative results on two retrieval benchmarks, we also provide a qualitative view of the uncertainty of the localization and extracted features. Lastly, we validate that the benefits of our cross-view matching approach on the retrieval benchmarks also translates to a realistic localization task using the particle filter and real GPS measurement data.

A. Datasets

We here discuss the two image retrieval benchmarks. While ideally the training data is collected to suit the application-specific r (see Sec. III-C.2), to reuse existing datasets in our experiments we instead assume a suitable target r value by considering each dataset's sample density.

CVACT Dataset: CVACT [3] is a large cross-view dataset with GPS footprint for image retrieval. It contains 35532 ground panorama and satellite image pairs, denoted as CVACT_train, and 92802 pairs as CVACT_test. Notably, the validation set CVACT_val of 8884 pairs is a subset of CVACT_test, and [4] reported their quantitative results on the CVACT_val rather than CVACT_test. We will not follow the data split in [3],[4], because CVACT_val is rather sparse, and it trivializes our task formulation of localization using a prior too much as it discarded all negative samples. Furthermore, we follow the target use-case where all satellite images are available during training and split only the ground images into training, validation, and test set. In total, there are 128334 satellite images, and the number of ground images is 86469, 21249, and 20616 in training, validation, and test set respectively. The data is relatively sparse: Using a localization prior of $r = 100$m, most samples have between 25 and 100 other pairs in their local neighborhood.

Oxford RobotCar Dataset: Oxford RobotCar [25],[26] is a dataset targeted at autonomous driving and contains images, raw GPS recordings, RTK measurements, etc., under different lighting and weather conditions collected in different times of the day and over a year in multiple traversals in the Oxford region.

The dataset has not been used for cross-view matching-based localization, as it does not contain satellite/aerial images. To construct a novel benchmark, we collected satellite images at zoom level 20 (~ 0.0924m per pixel) with the Google Maps Static API for each ground-level front-camera image. The satellite images were cropped into 600×600 pixel, which corresponds to a $55.44\text{m} \times 55.44\text{m}$ ground area, and the ground-level images are cropped to exclude visible parts of the ego-vehicle.

For now, we do not target the most extreme lighting and weather conditions and select the traversals recorded in different daytime and days with the label "sun", "overcast" or "clouds" and which contain both raw GPS and accurate RTK localization measurements. In the dataset, the front-viewing images are taken at 16Hz. To make sure the consecutive ground images do not look too similar in appearance, we subsample the images to make sure there is at least 5m between two consecutive frames in each traversal. Finally,

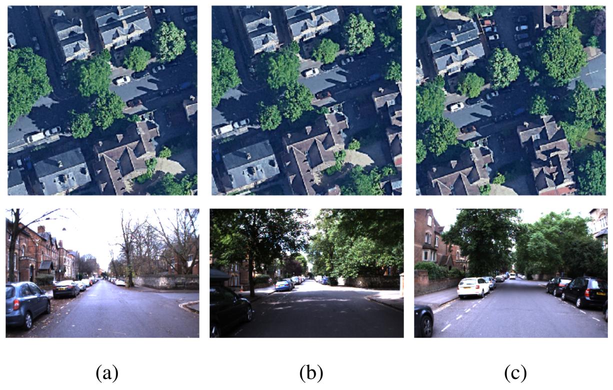


Fig. 4: Three sample pairs in the proposed Oxford RobotCar cross-view localization benchmark to highlight some local and global differences. (a) and (b) are 5m apart, (b) and (c) are 20m apart. Ground images are from different traversals and recording days, resulting in variations in cars, vegetation and lighting conditions.

we acquire the corresponding satellite images centered at the ground truth locations to build the ground-to-satellite pairs. In total, we obtain 23854 pairs from 15 traversals. We always keep all the satellite images and use the ground image from 11 traversals as the training set (17067), 1 traversal as the validation set (1698), 3 traversals as the test set (5089). Our chosen test traversals are collected in Summer (T.1,2) and Winter (T.3) with labels "overcast, roadworks" (T.1), "sun" (T.2), and "overcast" (T.3) to include variations in season, weather, and road conditions from the training set. For the same season and weather conditions, the test traversals are collected in later time-of-day than the training recordings. In this dense dataset, almost all images have more than 200 pairs in a $r = 50$m neighborhood. Some example ground and satellite pairs are shown in Figure 4.

In addition we also collect satellite images to cover the Oxford region at a grid with 5m interval, similar to [9]. This data will be used as the database $\mathbf{S}_{grid}$ for the particle filter of Sec. III-D to simulate a real-world localization task with the dataset's raw GPS and front-camera video stream.

B. Network architecture and implementation details

To implement the baseline SAFA method [4] we use the code released by its authors. For our method, we keep the network architecture equal and only replace the loss with our geo-distance weighted loss and train the model using local minibatches.[1] Both models are trained on our proposed data splits following the same procedure as in [4]: The VGG part is pre-trained on Imagenet [27], Adam [28] is used as optimizer with a learning rate of 10^{-5} on the CVACT dataset and $5 \cdot 10^{-5}$ on the Oxford RobotCar dataset. In the triplet losses, $\gamma = 10$, and the dropout keep rate is set to 0.8. On the traversal-based split Oxford RobotCar dataset, an additional dropblock [29] with a block size of 11 and keep probability of 0.8 is used to reduce overfitting.

[1]Our data (with an overview of time, season, label of chosen traversals) and code is available at `https://github.com/tudelft-iv/Visual-Localization-with-Spatial-Prior`

On the CVACT dataset where the ground images are 360° panoramic views, we use the polar transformed satellite images [4]. Due to its sparseness, the only correct match for a query ground image is the satellite patch centered the exact same location. On the dense Oxford RobotCar dataset, we observe that defining the training objective as matching the query to the satellite image at the exact location is too strict and the validation loss struggles to decrease. Therefore for a query ground image we select a random satellite image at a small geospatial offset of a maximum of 5m, which is the same distance used to subsample camera frames (see Sec. IV-A) As additional data augmentation, the satellite patches are also rotated by a random multiple of 90°.

C. Evaluation metrics

We will consider two aspects in our evaluation, namely image retrieval performance on the benchmarks, and localization performance for the particle filter.

For the retrieval, we assume at test time a known (worst-case) prior localization error of radius r, and thus directly discard for both methods any false negatives beyond r meters of the true location. Still, for reference, we also review the case when no such prior would be available (i.e. an *infinite* test radius). The recall@1 and recall@x meters are our quantitative metrics. They measure how often the top-1 retrieved satellite image is located at the exact location of, or less than x meters away from, the ground truth location. Although we introduced a maximal 5m geospatial offset in selecting matched satellite images for each query during training, we still report recall with $x < 5$m to give an overview of how top-1 retrieved satellite images distribute during testing.

As motivated in Sec I, recall does not reflect a model's localization performance. In the particle filter experiments, we instead measure the Euclidean localization error in meters between the true location and the median particle location during each traversal, and report the mean, median, 90%-quantile, 95%-quantile, and 99%-quantile error.

D. Effect of key hyperparameters

We test the impact of the three key hyperparameters: batch size N, weight decay d_r, and smoothness σ_{geo}.

We experimented with batch sizes $N = 4, 16, 64$. The batch size directly influences the training stability as it defines how many negative pairs are used when one positive pair is presented. On the Oxford RobotCar dataset, the training "collapsed" (i.e. descriptors are filled with only zeros) with small batches $N = 4$ / 16, around epoch 4 / 421 for our model, and around epoch 20 / 712 for the baseline. We find this behavior is due to values in the image descriptor (before normalization) exceeding numerical limits. Adding extra regularization does not prevent this. However, when $N = 64$, those values are kept under a much smaller magnitude. We reckon that when the batch size is too small with limited diversity in the satellite images, there is a risk that only maximizing the similarity between positive pairs is enough to push negative samples away in representation space, and the network will put very large weights on such similarities. Indeed, on the sparse but diverse CVACT dataset training does not collapse with $N = 4$ or $N = 16$ for either model. Unfortunately for $N = 64$ many locations will not have sufficient neighbors to fill the batch. We therefore keep $N = 16$ for CVACT, and $N = 64$ for Oxford RobotCar. In general, since r upper-bounds N, a small r requires dense training data to avoid potential training instability.

To choose between step decay and Gaussian decay for p_r, we kept other hyperparameters the same and trained our model with different decay options on the Oxford RobotCar dataset. The model trained with step decay surpasses the model trained with Gaussian decay by a large margin of 16.9% in recall@5m. We note that, for the same r, the Gaussian decay heavily down-weights far away samples, however these contribute greatly to the validation performance. We will therefore use $w_{geo}(i, j)$ with the step decay in the later experiments.

We also test $\sigma_{\text{geo}} = 0$, 5, 10, 15 meters, and find validation recall@5m is 75.5, 79.9, 82.5, 81.2 percent respectively on the Oxford RobotCar dataset ($\sigma_{\text{geo}} = 0$m indicates no down-weighting of nearby negative samples). Clearly, in this dataset where images are densely distributed, down-weighting the nearby negatives samples is important to learn a good representation. On the sparse CVACT dataset, we observe that σ_{geo} does not influence performance much. For the remainder, we fix $\sigma_{\text{geo}} = 10$m for both datasets.

E. Generalization across regions

The experiment on the CVACT dataset shows how well the learned representation generalizes to unseen ground images in new areas. Since locations are more sparsely distributed, we here use $r = 100$m as a weak hypothetical localization prior to train our model. To test the generality of geo-local training, we directly apply it to DSM [13] in addition to our regular SAFA baseline without further geo-local loss hyperparameter-tuning. All models are trained for 100 epochs, and we keep the best ones according to validation split performance.

We observe that geo-local models converge faster than the baselines even though the geo-distance weighted loss assigns zero weights to some triplets in the local minibatches. Evaluation results are reported on the test split in Table I. Providing the same localization prior to testing, our models improved the recall@1 by around 12.3% (74.0 vs 65.9 percent) for SAFA, and around 3.2% (70.4 vs 68.2 percent) for DSM. Meanwhile, our models also beat both baselines by a considerable margin in terms of the recall@5m and recall@10m. Importantly, these results confirm that our geo-local representation does not capture features that *identify* the local training region, which would not generalize, but captures features that *discriminate* nearby locations, which does generalize. Furthermore, the improvements of geo-local method generalize over different baselines, without the need for any baseline-specific hyperparameter tuning. As expected, globally (i.e. with ∞ test radius) our models perform worse than the baselines, as it violates the prior assumption. Still, in real-world applications, we do expect a coarse localization estimate to be present to benefit from geo-local features.

Recall@	1	1	5m	10m
Test Radius	100m	∞	100m	100m
SAFA-local (%)	**74.0**	55.8	**77.5**	**85.4**
SAFA[4] (%)	65.9	**59.9**	68.8	78.2
DSM-local (%)	**70.4**	56.6	**73.9**	**82.0**
DSM[13] (%)	68.2	**64.0**	71.5	80.4

TABLE I: Evaluation on CVACT Test Set (best in bold). The term “local” means the model trained with our geo-local loss.

F. Generalization across time

On the Oxford RobotCar dataset, we test how well the learned representation generalizes to new ground images collected on other dates and different times of the day in the same region. Since the images are distributed much denser here, we use a more realistic hypothetical localization prior, $r = 50$m. The best model is kept according to the validation performance in 1000 epochs of training.

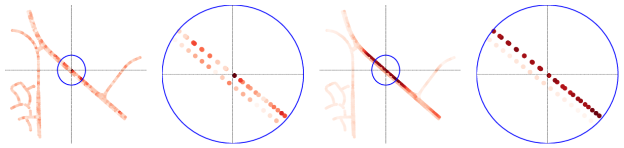
Recall@	1	1m	3m	5m	5m
Test Radius	50m	50m	50m	50m	∞
T.1 our model (%)	**9.7**	**38.4**	**84.5**	**96.0**	64.3
T.1 baseline[4] (%)	7.1	26.3	75.5	92.3	**90.6**
T.2 our model (%)	**8.3**	**29.6**	**71.8**	**85.9**	43.0
T.2 baseline[4] (%)	5.3	19.5	59.4	81.9	**76.1**
T.3 our model (%)	**8.4**	**28.9**	**77.2**	**88.9**	53.0
T.3 baseline[4] (%)	5.7	21.4	62.0	83.4	**79.5**
Mean our model (%)	**8.8**	**32.3**	**77.8**	**90.3**	53.4
Mean baseline[4] (%)	6.0	22.4	65.6	85.9	**82.1**

TABLE II: Evaluation on Oxford RobotCar Test Sets (best results in bold). T.N stands for the Nth testing traversal, and the mean recall over 3 traversals is in the bottom row.

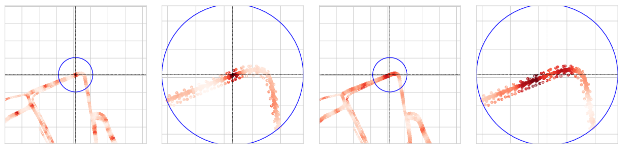
The quantitative test results of the selected model are summarized in Table II. When the localization prior is available, our learned geo-local representation consistently outperforms the baseline on all test traversals. For completeness, we note again that the baseline outperforms our model for global localization. Overall, our approach generalizes well across time-of-day and different days, and it does not overfit on the training ground images or specific time and weather conditions, which is important as in practice localization in the target region will be done on different days.

G. Qualitative results

In this section, we try to illustrate how our model performs differently from the baseline. To provide a qualitative view of the model behavior, we visualize the localization heat map using the similarity measurement between a test query and all nearby satellite images. On both CVACT, Figure 5a, and Oxford RobotCar, Figure 5b, our model outputs a sharper localization result inside the prior area, while the baseline has more uncertainty about the exact location along the road. Unlike the baseline, our method also produces other high-probability peaks outside the circle. This is because it does not distinguish distinct areas, and similar local spatial layouts may reoccur elsewhere. This trade-off comes from the geographically local representation our model uses.

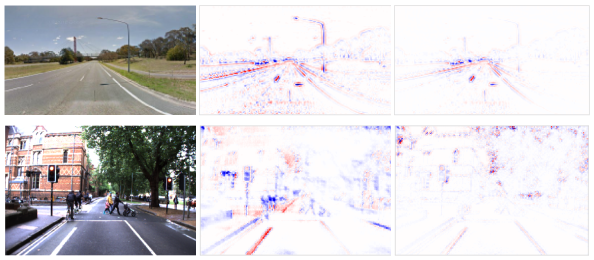


(a) Zoomed out/in views of localization heat map on CVACT dataset (left: ours, right: baseline)



(b) Zoomed out/in views of localization heat map on Oxford RobotCar dataset (left: ours, right: baseline)

Fig. 5: Examples of localization heat maps on (a) CVACT and (b) Oxford RobotCar dataset. Each dot represents a satellite image with the darkness proportional to the similarity to the query image. The ground truth location of the query is indicated by the cross. The circle indicates the local neighborhood with radius $r = 100$m in (a) and $r = 50$m in (b). The zoomed-out image shows the surrounding 1km $\times$ 1km area in (a) and 400m $\times$ 400m area in (b). On both datasets, our approach results in a single peak within the local neighborhood, while the baseline has more uncertainty.



(a) Input (b) Ours (c) Baseline

Fig. 6: Visualized back-propagated encoded feature attention maps for a ground image in CVACT dataset (first row) and Oxford RobotCar dataset (second row).

We can also verify this by comparing the encoded image features of both approaches. Similar to [4], we back-propagate the spatial embedding maps to the input image to show where the model extracts features [30], see Figure 6. On the CVACT dataset our model pays attention to vegetation and streetlights. The baseline model, on the other hand, ignores these objects and focuses on the road structure. On the Oxford RobotCar dataset our model looks for traffic lights and building facades, while the baseline mostly looks at the canopies and building roofs. The objects our model pays attention to are repeated at many different places, nevertheless they are useful in disambiguating other images along this road. The baseline focuses on fewer environmental details, which are sufficiently discriminative globally but not locally.

H. Temporal filtering

Finally, we validate that the better performance of our model in the discussed benchmarks also translates to actual gains in a real-world localization task using actual GPS measurements and temporal filtering priors, as opposed to hypothetical priors. The localization pipeline is tested on the Oxford RobotCar dataset with a update rate of 1.6Hz, where every 10th image from the unsampled test traversals is used as our ground-level query and being matched to regularly distributed satellite images. We do not include additional sensors in the temporal filter, such as wheel odometry and IMU, to keep the amount of tuneable system configurations and parameters to a minimum.

The quality of the GPS measurements controls the hyperparameter σ_{gps}. Unfortunately, the GPS error is often unpredictable and can vary significantly. For example, the mean error of the raw GPS positioning on Oxford RobotCar test traversals is around 3.7m, but reaches 13m on the validation traversal. In our experiment, we set σ_{gps} to 10m.

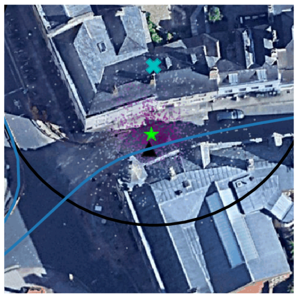
Localization error(m)	mean	50%	90%	95%	99%
T.1 ours+GPS	**2.65**	**2.12**	**4.70**	**5.91**	**11.00**
T.1 baseline[4]+GPS	3.23	2.63	5.71	7.27	14.91
T.1 GPS	4.66	3.93	8.24	10.73	20.89
T.2 ours+GPS	**2.73**	**2.46**	**4.71**	**5.73**	**8.12**
T.2 baseline[4]+GPS	3.19	2.71	5.58	7.02	11.90
T.2 GPS	4.50	4.00	7.48	9.28	19.19
T.3 ours+GPS	**2.94**	**2.49**	**5.46**	**6.92**	**10.80**
T.3 baseline[4]+GPS	3.53	2.72	6.69	8.30	14.69
T.3 GPS	4.64	3.92	8.76	10.64	20.51
Mean ours+GPS	**2.77**	**2.36**	**4.96**	**6.19**	**9.97**
Mean baseline[4]+GPS	3.32	2.69	5.99	7.53	13.83
Mean GPS	4.60	3.95	8.16	10.22	20.20

TABLE III: Particle filter localization error (mean and error at x%-quantile) on Oxford RobotCar test traversals. Best results in bold. "ours+GPS" and "baseline+GPS" use both the cross-view matching module and GPS. "GPS" is without any cross-view matching.

The quantitative results[2] over 3 test traversals are summarized in Table III. Temporal filtering of raw GPS alone in the particle filter achieves an average error of $\sim$ 5m. Incorporating the cross-view matching improves localization significantly, especially when GPS produces spurious large outliers, as seen from the 99%-quantile error. An example is shown in Figure 7. Importantly, our model delivers overall the best accuracy and robustness, and reduces mean (2.77 vs 3.32m) by 17% and 99%-quantile error (9.97 vs 13.83m) by 28% compared to the baseline.

The superiority of our method together with GPS and particle filter comes from the sharp cross-view matching result. Most of the time, using GPS is enough for global coarse localization, and adding another coarse estimate from

[2]We include a video of localization results in our supplementary material


(a) Localization result (ours)

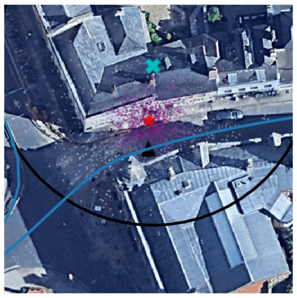

(b) Localization result (baseline)

Fig. 7: Particle filter-based localization. Each purple dot (particle) has a darkness (re-sampling weight). The cyan cross and black circle shows the raw GPS positioning and its 95% confidence interval. The black triangle marks the ground truth location on the full trajectory (blue line). The green (red) star is the localization result by using our model (the baseline) in the pipeline.

global cross-view matching does not gain much in localization accuracy. In contrast, our method effectively refines the GPS positioning within GPS-uncertain areas. Extreme erroneous GPS measurements are filtered out by the outlier rejection module in the temporal filter, ensuring a reasonable prior is obtained from previous time instances.

Note that, in other regions where there are many high rising buildings, a larger σ_{gps} could give better localization results. However, we observed on our validation traversal that also for different σ_{gps} values in the range from 5m to 30m our model still outperforms the baseline, and does not influence our conclusion here.

V. Conclusions

In this work, we embedded the cross-view matching into a workable real-world localization system by considering the prior from other localization components in the training. We quantitatively and qualitatively showed the advantage of geo-local training over geo-global training on state-of-the-art methods. A 12.3% improvement of the recall@1 was achieved on the CVACT dataset. On the Oxford RobotCar dataset, we improved the recall@5m from 85.9% to 90.3%. Besides, we also demonstrated that the increase in cross-view matching capability translates to 17% lower mean and 28% lower 99%-quantile localization error when real GPS measurements and cross-view matching scores are fused in a particle filter-based localization pipeline. More importantly, all noticeable quantitative benefits come from a simple to implement and generic adaptation. Future work can test how our model generalizes across day-and-night, weather, and season by strictly control each factor in the training and test split with a more comprehensive dataset.

ACKNOWLEDGMENT

This work is part of the research programme Efficient Deep Learning (EDL) with project number P16-25, which is (partly) financed by the Dutch Research Council (NWO).

References

[1] 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. of IEEE CVPR*, 2018, pp. 7258–7267.
[2] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *Proc. of IEEE CVPR*, 2015, pp. 5007–5015.
[3] L. Liu and H. Li, "Lending orientation to neural networks for cross-view geo-localization," in *Proc. of IEEE CVPR*, 2019, pp. 5624–5633.
[4] Y. Shi, L. Liu, X. Yu, and H. Li, "Spatial-aware feature aggregation for image based cross-view geo-localization," in *NeurIPS*, 2019, pp. 10 090–10 100.
[5] N. N. Vo and J. Hays, "Localizing and orienting street views using overhead imagery," in *ECCV*. Springer, 2016, pp. 494–509.
[6] S. Workman, R. Souvenir, and N. Jacobs, "Wide-area image geolocalization with aerial reference imagery," in *Proc. of IEEE ICCV*, 2015, pp. 3961–3969.
[7] B. Ben-Moshe, E. Elkin, *et al.*, "Improving accuracy of gnss devices in urban canyons," in *CCCG*, 2011, pp. 511–515.
[8] E. Kaplan and C. Hegarty, *Understanding GPS: principles and applications*. Artech house, 2005.
[9] S. Hu and G. H. Lee, "Image-based geo-localization using satellite imagery," *IJCV*, pp. 1–15, 2019.
[10] L. Heng, B. Choi, Z. Cui, *et al.*, "Project autovision: Localization and 3d scene perception for an autonomous vehicle with a multi-camera system," in *ICRA*, 2019, pp. 4695–4702.
[11] J. Martinez, S. Doubov, J. Fan, I. A. Bârsan, S. Wang, G. Máttyus, and R. Urtasun, "Pit30m: A benchmark for global localization in the age of self-driving cars," in *IEEE/RSJ IROS*, 2020, pp. 4477–4484.
[12] M. Zhai, Z. Bessinger, S. Workman, and N. Jacobs, "Predicting ground-level scene layout from aerial imagery," in *Proc. of IEEE CVPR*, 2017, pp. 867–875.
[13] Y. Shi, X. Yu, D. Campbell, and H. Li, "Where am i looking at? joint location and orientation estimation by cross-view matching," in *Proc. of IEEE CVPR*, 2020, pp. 4064–4072.
[14] Y. Shi, X. Yu, L. Liu, *et al.*, "Optimal feature transport for cross-view image geo-localization," in *Proc. of AAAI*, 2020, pp. 11 990–11 997.
[15] Z. Xia, O. Booij, M. Manfredi, and J. F. Kooij, "Geographically local representation learning with a spatial prior for visual localization," in *ECCV Workshops*. Springer, 2020, pp. 557–573.
[16] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proc. of IEEE CVPR*, 2016, pp. 5297–5307.
[17] R. Arandjelovic and A. Zisserman, "All about VLAD," in *Proc. of IEEE CVPR*, 2013, pp. 1578–1585.
[18] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. of IEEE CVPR*, 2010, pp. 3304–3311.
[19] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.
[20] K. Regmi and M. Shah, "Bridging the domain gap for ground-to-aerial image matching," in *Proc. of IEEE ICCV*, 2019, pp. 470–479.
[21] A. Toker, Q. Zhou, M. Maximov, and L. Leal-Taixé, "Coming down to earth: Satellite-to-street view synthesis for geo-localization," *arXiv preprint:2103.06818*, 2021.
[22] S. Zhu, T. Yang, *et al.*, "Vigor: Cross-view image geo-localization beyond one-to-one retrieval," *arXiv preprint:2011.12172*, 2020.
[23] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.
[24] S. Zair, S. L. Hégarat-Mascle, and E. Seignez, "Coupling outlier detection with particle filter for gps-based localization," in *IEEE ICITS*, 2015, pp. 2518–2524.
[25] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *IJRR*, vol. 36, no. 1, pp. 3–15, 2017.
[26] W. Maddern, G. Pascoe, *et al.*, "Real-time kinematic ground truth for the oxford robotcar dataset," *arXiv preprint: 2002.10152*, 2020.
[27] J. Deng, W. Dong, R. Socher, *et al.*, "Imagenet: A large-scale hierarchical image database," in *Proc. of IEEE CVPR*, 2009, pp. 248–255.
[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, 2014.
[29] G. Ghiasi *et al.*, "Dropblock: A regularization method for convolutional networks," in *NeurIPS*, 2018, pp. 10 727–10 737.
[30] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*. Springer, 2014, pp. 818–833.