MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

VLASE: Vehicle Localization by Aggregating Semantic Edges

Yu, Xin; Chaturvedi, Sagar; Feng, Chen; Taguchi, Yuichi; Lee, Teng-Yok; Fernandes, Clinton; Ramalingam, Srikumar

TR2018-113 August 17, 2018

Abstract

We propose VLASE, a framework to use semantic edge features from images to achieve on-road localization. Semantic edge features denote edge contours that separate pairs of distinct objects such as building-sky, road-sidewalk, and building-ground. While prior work has shown promising results by utilizing the boundary between prominent classes such as sky and building using skylines, we generalize this to consider 19 semantic classes. We extract semantic edge features using CASENet architecture and utilize VLAD framework to perform image retrieval. We achieve improvement over state of-the-art localization algorithms such as SIFT-VLAD and its deep variant NetVLAD. Ablation study shows the importance of different semantic classes, and our unified approach achieves better performance compared to individual prominent features such as skylines. We also introduce SLC Marathon dataset, a challenging dataset covering most of Salt Lake City with sufficient lighting variations.

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

VLASE: Vehicle Localization by Aggregating Semantic Edges

Xin Yu[1*], Sagar Chaturvedi[1*], Chen Feng[2], Yuichi Taguchi[3], Teng-Yok Lee[3], Clinton Fernandes[1], Srikumar Ramalingam[1]

***Abstract*— We propose VLASE, a framework to use semantic edge features from images to achieve on-road localization. Semantic edge features denote edge contours that separate pairs of distinct objects such as building-sky, road-sidewalk, and building-ground. While prior work has shown promising results by utilizing the boundary between prominent classes such as sky and building using skylines, we generalize this to consider 19 semantic classes. We extract semantic edge features using CASENet architecture and utilize VLAD framework to perform image retrieval. We achieve improvement over state-of-the-art localization algorithms such as SIFT-VLAD and its deep variant NetVLAD. Ablation study shows the importance of different semantic classes, and our unified approach achieves better performance compared to individual prominent features such as skylines. We also introduce SLC Marathon dataset, a challenging dataset covering most of Salt Lake City with sufficient lighting variations.**

I. Introduction

In the pre-GPS era, location was not specified in latitude-longitude coordinates. The typical description of a location is based on certain semantic proximity, such as a tall building, traffic light, or an intersection. While the recent image-based localization methods rely on either complex hand-crafted features like SIFT [1] or automatically learnt features using CNNs, we would like to take a step back and ask the following question: How powerful are simple semantic cues for the task of localization? There is a general consensus that the salient features for localization are not always human-understandable, and it is important to capture special visual signatures imperceptible to the eye. We show that simple human-understandable semantic features, although extracted using CNNs, provide accurate localization in urban scenes and they compare favorably to some of the state-of-the-art localization methods that employ SIFT features in a VLAD [2] framework.

Figure 1 illustrates the basic idea of this paper. Given an image from a vehicle, we first detect semantic boundaries, the pixels between different object classes. In this paper, we use the recently introduced CASENet [4] architecture to extract semantic boundaries. The CASENet architecture provides a multi-label framework where the edge pixels are associated with more than one object classes. For example, a pixel lying on the edge between sky and buildings will be associated with both sky and building class labels. This allows our method to unify multiple semantic classes as

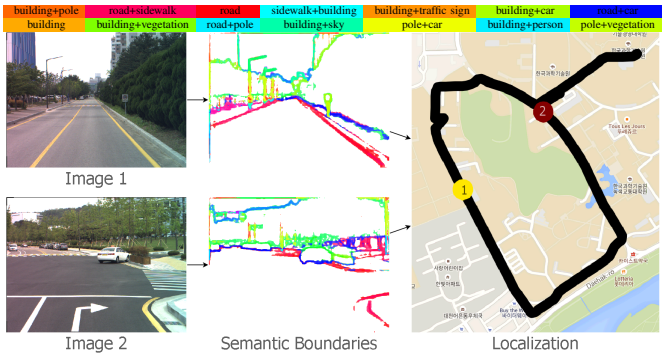


Fig. 1. Illustration of VLASE. Given images (left) from a vehicle, we extract semantic edge features (middle). Different colors indicate different combinations of object classes. The extracted semantic features are compared to the features from geo-tagged images in a database to estimate the location. In this example, the red and yellow circles on the map (right) indicate the locations of the two given images. (The images are from the KAIST WEST sequences captured at 9AM [3].).

localization features. The middle column of Figure 1 shows the semantic edge features. By matching the semantic edge features between a query image and geo-tagged images in a database, which is achieved using VLAD in this paper, we can estimate the location of the query image, as illustrated on the Google map in the right of Figure 1.

Besides the matching between semantic edge features, we also observed that in the context of on-road vehicles, appending 2D spatial location information with the extracted features (SIFT or CASENet) boosts the localization performance by a large margin. In this paper, we heavily rely on the prior that the images are captured from a vehicle-mounted camera, and exploit edge features that are typical in urban scenes. In addition, we sample only a very limited set of poses for on-road vehicles. The motion is near-planar and the orientation is usually aligned with the direction of the road. It is common to make this assumption for accurate vehicle localization in urban canyons, where GPS suffers from multi-path effects.

We briefly summarize our contributions as follows:

- We propose VLASE, a simple method that uses semantic edge features for the task of vehicle localization. While prior methods use individual features such as horizon, road maps, and skylines [5]–[8], we propose a unified framework that allows the incorporation of multiple semantic classes.
- We show that it is beneficial to augment semantic features by 2D spatial coordinates. This is counter-intuitive to prior methods that utilize invariant features in a bag-of-words paradigm. In particular, we show

*indicate equal contributions.
[1]University of Utah, Salt Lake City, UT 84112, USA {xin.yu,sagar.chaturvedi,srikumar}@utah.edu
[2]New York University, Brooklyn, NY 11201, USA cfeng@nyu.edu
[3]Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA {taguchi,tlee}@merl.com

that even standard SIFT-VLAD can be significantly improved by embedding additional keypoint locations.
- We show compelling experimental results on two different datasets, including the public KAIST [3] and a route collected by us in Salt Lake City. We outperform competing localization methods such as standard SIFT-VLAD [2], pre-trained NetVLAD [9], and the coarse localization in [10], even with smaller descriptor dimensions.
- We will release SLC Marathon dataset, a new challenging dataset covering most of Salt Lake City with significant lighting variations.

II. Related Work

The vision and robotics communities have witnessed the rise of accurate and efficient image-based localization techniques that can be complementary to GPS, which is prone to error due to multi-path effects [11]. The techniques can be classified into regression-based methods and retrieval-based ones. Regression-based methods [12]–[14] directly obtain the location coordinates from a given image using techniques such as CNNs. Retrieval-based methods match a given query image to thousands of geo-tagged images in a database, and predict the location estimates for the query image based on the nearest or k-nearest neighbors in the database. Regression-based methods provide the best advantage in both memory and speed. For example, methods like PoseNet [12] do not require huge database with millions of images and the location estimation can be done in super-real time (e.g. 200 Hz). On the contrary, retrieval-based ones are usually slower and have a large memory requirement for storing images or its descriptors for the entire city of globe. However, the retrieval-based methods typically provide higher accuracy and robustness [15].

A. Features

In this paper, we will focus on the retrieval-based approach, which essentially finds the distance between a pair of images using extracted localization features. Based on human understandability, we broadly classify the localization features into the following two categories:

Simple Features: We refer to simple features as the ones that are human-understandable: line-segments, horizon, road maps, and skylines. Skylines or horizon separating sky from buildings or mountains can be used for localization [5]–[8]. Several existing methods use 3D models and/or omni-directional cameras for geolocalization [6], [16]–[23]. Line segments have been shown to be very useful for localization. The localization can be achieved by registering an image with a 3D model or a geo-tagged image. By directly aligning the lines from query images to the ones in a line-based 3D model we can achieve localization [16], [24], [25], [37]. Furthermore, even simply aligning high gradient pixels can be extremely beneficial for visual odometry tasks [34]. Semantic segmentation of buildings has been used for registering images to 2.5D models [26]. Other simple localization features include travel time stamps [36].

We can also use other human-understandable simple feature such as roadmaps or weather patterns to obtain localization. Visual odometry can provide the trajectory of a vehicle in motion, and by comparing this with the roadmaps, we can compute the location of the vehicle [27], [28]. It is intriguing to see that even weather patterns can act as signatures for localizing an image [29].

Complex Features: The complex ones are visual patterns extracted through hand-crafted feature descriptors or automatically extracted ones using CNNs. These class of features are referred to as complex ones since they are not human-understandable, i.e, not easily perceptible to human eye. It is possible to achieve localization in a global scale using GPS-tagged images from the web and matching the query image using a wide variety of image features such as SIFT, SURF, and ORB [30]–[33].

The use of neural networks for localization is an old idea. RATSLAM [38] is a classical SLAM algorithm that uses a neural network with local view cells to denote locations and pose cells to denote heading directions. The algorithm produces "very coarse" trajectory in comparison to existing SLAM techniques that employ filtering methods or bundle-adjustment machinery. Kendall et al. [12] presented PoseNet, a 23 layer deep convolutional neural network based on GoogleNet [39], to compute the pose in a large-region at 200 Hz. CNN can be also applied to learn the distance metric to match two images. As one can achieve localization by matching an image taken at the ground level to reference database of geo-tagged bird's eye, aerial, or even satellite images [40]–[43], such cross-matching is typically done using siamese networks [44]. Recently, it was shown that LSTMs can be used to achieve accurate localization in challenging lighting conditions [45]. A survey of different state-of-the-art localization techniques is given in [46], and there has been releases of many newer datasets [15], [47]. The idea of dominant set clustering is powerful for localization tasks [48]. Many existing methods formulate localization problem in a similar manner to per-exemplar SVMs in object recognition. To handle the limitation of having very few positive training examples, a new approach to calibrate all the per-location SVM classifiers using only the negative examples is proposed [49].

B. Vocabulary tree

In the retrieval based methods, we match a query image to millions of images in a database. The computation efficiency is largely addressed by bag-of-words (BOW) representation that aggregates local descriptors into a global descriptor, and enables fast large-scale image search [51]–[53]. Recently, extensions of BOW including the Fisher vector and Vector of Locally Aggregated Descriptors (VLAD) showed state-of-the-art performance [2]. Experimental results demonstrate that VLAD significantly outperforms BOW for the same size. It is cheaper to compute and its dimensionality can be reduced to a few hundreds of components by PCA without noticeably impacting its accuracy.

The logical extension to VLAD is NetVLAD, which mimics VLAD in a CNN framework using a trainable generalized VLAD layer, NetVLAD, for the place recognition task [9]. This layer can be used in any CNN architecture and allows training via backward propagation. NetVLAD was shown to outperform non-learnt image representations and off-the-shelf CNN descriptors on two challenging place recognition benchmarks, and improves over current state-of-the-art compact image representations on standard image retrieval benchmarks.

In this paper, we combine the above two categories by localizing from human-interpretable semantic edge features learnt from a state-of-the-art CNN [4]. Note that very recently semantic segmentation is also used with either a sparse 3D model [10] or depth images [50] for long-term 3D localization. We show by experiments that VLASE improves the semantic-histogram-based coarse localization in [10].

III. Semantic Edges for Localization

This section explains our main algorithm of using semantic edge features for localization. The main idea is very simple. Similar to the use of SIFT features in a VLAD framework, we use CASENet multi-label semantic edge class probabilities as compact, low-dimensional, and interpretable features. Similar to standard BOW, VLAD also constructs a codebook from a databse of feature descriptors (SIFT or CASENet) by performing a simple K-means clustering algorithm on those descriptors. Here we denote M clusters as $\mathcal{C} = \{c_1, \ldots, c_M\}$. Given a query image, each of its feature descriptors x_i is associated to the nearest cluster c_j in the codebook. The main idea in VLAD is to accumulate the difference vector $x_i - c_j$ for every x_i that is associated with c_j. VLAD is considered to be superior to traditional BOW methods mainly because this residual statistic provides more information and enables better discrimination.

To detect the semantic edges, we use the recently introduced CASENet architecture, whose code is publicly available [1]. Given an input image $\mathbf{I}$, we first apply a pretrained CASENet to compute the multi-label semantic edge probabilities $\mathbf{Y}(\mathbf{p}) = [\mathbf{Y}_1(\mathbf{p}), \cdots, \mathbf{Y}_K(\mathbf{p})]$ for each pixel $\mathbf{p} \in \mathbf{I}$. Here K is the number of object classes. Then we select all edge pixels $\{\mathbf{q} \in \mathbf{I} | \mathbf{Y}_k(\mathbf{q}) \geq T_e, \exists k \in [1, \cdots, K]\}$, i.e., pixels that have at least one semantic edge label probability exceeding a given threshold T_e. Thus, for any image, we can compute a set of K-dimensional CASENet edge features (for the Cityscapes dataset, $K = 19$). We further augment this K-dimensional feature by appending a 2-dimensional normalized-pixel-position feature $[q_x/W, q_y/H]$, where W and H are the fixed image width and height, and q_x and q_y are the column and row index respectively for a pixel $\mathbf{q}$. We will refer to such a $K+2$ dimensional feature $\hat{\mathbf{Y}}$ as an augmented CASENet edge feature.

Due to the often much larger number of edge pixels compared to SIFT/SURF features in an image, to build a visual codebook or vocabulary following the VLAD framework, we run a sequential instead of a full K-means algorithm (MiniBatchKMeans, implemented in the python package scikit-learn) using all the augmented CASENet edge features on one training image as a mini-batch. This is iterated over the whole training image set for multiple epochs until it converges to M centers, each in the $K+2$ dimensional space, to form the trained CASENet-VLAD codebook. An example is visualized in Figure 2.

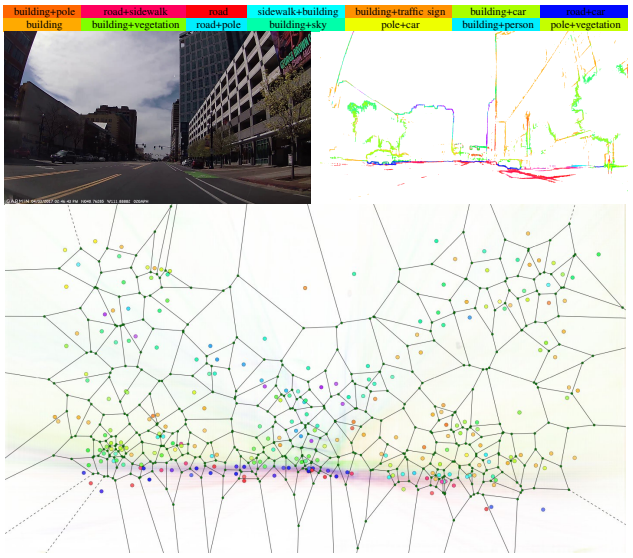


Fig. 2. CASENet edge feature and VLAD. Top: An example input image (left) and its CASENet features (right). Each color corresponds to an object class. Bottom: Visualization of a CASENet-VLAD vocabulary of $M = 256$ codewords/cluster centers, shown as color-coded dots. For the dot of each cluster, its x-y positions correspond to $\hat{\mathbf{Y}}_{20}$ and $\hat{\mathbf{Y}}_{21}$, and its color is computed from CASENet features $\mathbf{Y}_k$. The Voronoi graph (black edges with small green nodes) shows the CASENet-VLAD division of the x-y image space. The background of the bottom image is an average of CASENet feature visualization from all images used to train the codebook. As the background shows an averaged semantic on-road driving scene, it can be seen that the colors of the dots in the cluster centers distribute similarly to the colors of this average scene.

To perform on-road place recognition, we first need to process a sequence of images serving as the visual map, i.e., the mapping sequence. This can be simply done by extracting all augmented CASENet edge features on each image and compute a corresponding $M \times (K+2)$ CASENet-VLAD descriptor $\mathbf{D}$ using the trained codebook, with power-normalization followed by L2-normalization. The CASENet-VLAD descriptors for the mapping sequence are then stored in a database. During place recognition, we repeat this process for the current query image to get its CASENet-VLAD descriptor and search in the mapping database for the top-N most similar descriptors using cosine-distance. This pipeline is further illustrated in Figure 3.

IV. Datasets

We have experimented on 3 visual place recognition datasets. The first two are called SLC Urban and SLC Marathon, which were captured in Salt Lake City downtown. The third is called KAIST, which is one of the routes from the KAIST All-Day Visual Place Recognition dataset [3].

[1] http://www.merl.com/research/license#CASENet

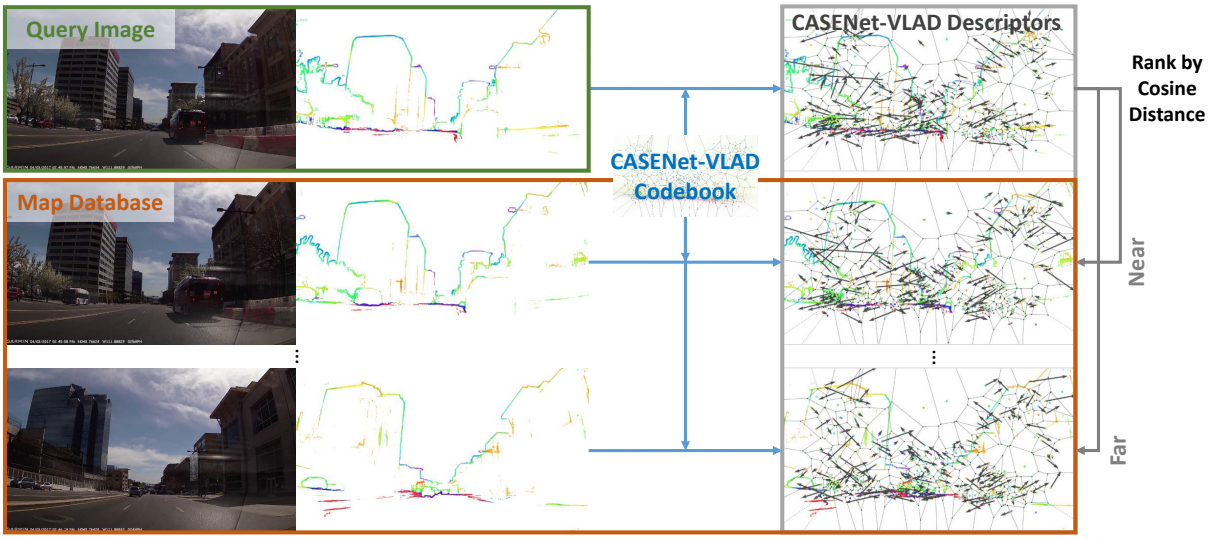
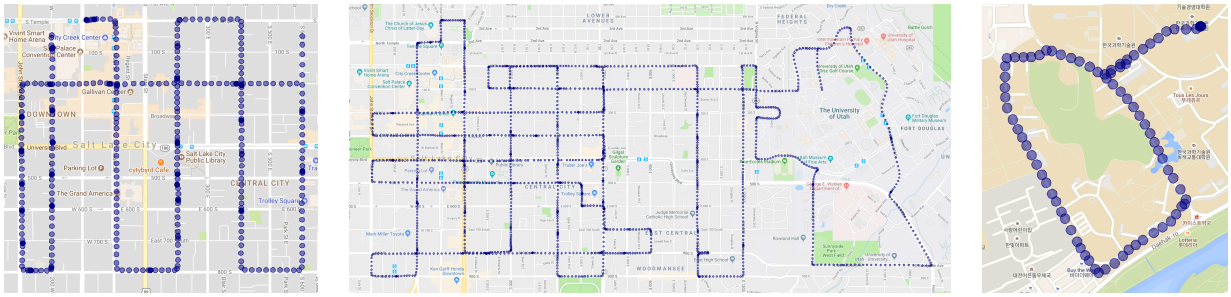


Fig. 3. VLASE pipeline. All mapping images are first processed by CASENet, from which we can build a VLAD codebook using all CASENet features. We then compute each image's CASENet-VLAD descriptor $\mathbf{D}$ (the last two dimensions of each residual vector, i.e., $\mathbf{D}(:,20)$ and $\mathbf{D}(:,21)$, are visualized as 2D vectors origin at the corresponding codeword/cluster center, i.e., $\mathbf{C}_m$). During localization, we similarly compute the currently observed image's CASENet-VLAD descriptor, and query in the database for the top-N closest descriptors in terms of cosine distance. Note that while the geometry shape of the three CASENet edges in column two are visually similar to each other, their corresponding CASENet-VLAD descriptors in the last column are more discriminative, even only visualized by the last two dimensions.



SLC Urban SLC Marathon KAIST West

Fig. 4. The testing routes of our experiments.

A. SLC

We created our SLC Urban dataset by capturing two video sequences in the downtown of Salt Lake City, with abundance of objects belonging to the classes in the Cityscapes dataset. The route length is about 15km, which is shown in Figure 4 left. We used a Garmin dash-cam to collect videos of the scenes in front of the vehicle. This dash-cam stored the videos at 30 FPS, and the two sequences have 98513 and 89633 frames, which are captured between 15:00-17:00 continually. We resized the image from the original resolution 1920×1080 to 640×360 pixels. A special feature of this dash-cam is that it also encodes the GPS coordinates in latitude and longitude, which provides the ground truth of our video frames. Since the frame rate of SLC sequence is 30 fps but only the first frame within every second has a GPS coordinate, we sampled every 30 frames from each sequence. We use the longer one as a database of 3284 images and compute the VLAD codebook. The other sequence has 2988 sampled frames for querying. Similarly, we created a larger dataset, SLC Marathon (Figure 4 middle): route length 46km, 24156 images as the database (captured between 19:00-21:00), and 11663 images for querying (between 14:30-16:30). Note that the sequences in this dataset were captured at different times, and thus they have adequate lighting variations for same locations, making it more challenging for localization.

B. KAIST

The KAIST dataset was captured by Choi et al. [3] in the campus of Korea Advanced Institute of Science and Technology (KAIST). They captured 42 km sequences at 15-100Hz using multiple sensor modalities such as fully aligned visible and thermal devices, high resolution stereo visible cameras, and a high accuracy GPS/IMU inertial navigation system. The sequences covered 3 routes in the campus, which are denoted as west, east and north. Each route has 6 sequences recorded at different times of a day, including day (9 AM, 2 PM), night (10 PM, 2 AM), sunset (7 PM), and sunrise (5 AM). As these sequences capture various illumination conditions, this dataset is helpful for benchmarking under lighting variations.

TABLE I

ABLATION STUDY RESULTS ARE SHOWN IN THE FORMAT (a, b), WHERE a DENOTES SLC URBAN AND b DENOTES SLC MARATHON.

Urban,Marathon	Top-1 Accuracy			Top-5 Accuracy		
Removed	**5m**	**10m**	**20m**	**5m**	**10m**	**20m**
Road	53, 26	79, 48	91, 70	89, 45	96, 65	98, 80
Sidewalk	54, 26	80, 47	91, 68	87, 45	94, 64	96, 79
Building	50, 24	75, 44	86, 63	81, 41	90, 59	92, 76
Wall	50, 26	76, 47	87, 69	85, 45	92, 64	94, 80
Fence	54, 26	80, 48	90, 69	87, 45	94, 64	96, 80
Pole	51, 23	75, 44	88, 63	85, 42	93, 62	95, 75
Light	51, 25	75, 47	87, 68	84, 46	92, 65	95, 80
Sign	51, 26	76, 48	87, 69	85, 45	93, 64	95, 80
Veg	50, 24	74, 44	85, 63	83, 43	92, 61	95, 75
Terrain	51, 26	77, 47	88, 68	84, 44	91, 64	94, 80
Sky	50, 24	75, 43	85, 63	82, 41	91, 60	93, 76
Person	52, 27	79, 49	90, 71	87, 46	94, 65	96, 81
Rider	51, 26	78, 47	89, 69	87, 45	94, 64	96, 80
Car	54, 27	82, 48	93, 70	89, 46	97, 65	98, 82
Truck	53, 26	80, 48	91, 69	88, 45	95, 65	97, 80
Bus	51, 26	77, 47	88, 68	84, 44	92, 65	95, 79
Train	54, 26	79, 48	91, 69	87, 45	95, 64	97, 80
Motorcycle	51, 26	77, 48	88, 69	85, 45	92, 64	95, 80
Bicycle	52, 26	77, 47	89, 69	86, 44	94, 64	96, 80
Combinations	**5m**	**10m**	**20m**	**5m**	**10m**	**20m**
All	52, 26	78, 47	90, 69	87, 45	94, 64	96, 80
Static	56, 26	82, 47	94, 67	91, 44	98, 64	99, 80
Bld-Sky	49, 15	73, 30	85, 39	77, 28	91, 43	94, 54
Veg-Sky	57, 19	83, 38	95, 56	89, 35	96, 52	98, 70
Veg-Bld-Sky	55, 23	80, 44	91, 64	86, 41	94, 61	96, 79
All w/o (x,y)	44, 13	67, 23	76, 33	77, 26	86, 38	89, 50
Baselines	**5m**	**10m**	**20m**	**5m**	**10m**	**20m**
SIFT+(x,y)	32, 12	47, 22	60, 34	48, 25	61, 39	66, 52
SIFT	22, 1	36, 3	43, 6	32, 3	45, 6	48, 10
Toft [10]	32, 8	55, 16	63, 22	57, 17	73, 28	79, 36

TABLE II

ABLATION STUDY RESULTS FOR THE KAIST DATASET.

	Top-1 Accuracy			Top-5 Accuracy		
Removed	**5m**	**10m**	**20m**	**5m**	**10m**	**20m**
Road	72	84	90	88	91	94
Sidewalk	71	84	91	88	92	95
Building	71	84	90	88	91	94
Wall	73	85	90	87	91	94
Fence	73	86	92	90	93	96
Pole	70	84	89	87	91	94
Light	73	86	91	88	93	95
Sign	71	84	90	88	92	95
Veg	69	82	87	87	91	93
Terrain	72	84	90	88	91	94
Sky	73	85	91	88	93	95
Person	74	86	91	89	92	95
Rider	72	85	90	88	92	95
Car	77	88	93	91	94	96
Truck	72	86	90	89	93	94
Bus	74	86	90	89	92	94
Train	74	85	91	88	92	95
Motorcycle	72	85	90	88	92	95
Bicycle	73	85	90	88	92	95
Combinations	**5m**	**10m**	**20m**	**5m**	**10m**	**20m**
All	73	85	91	89	92	95
Static	77	88	92	91	94	96
Bld-Sky	62	74	83	82	87	91
Veg-Sky	73	83	88	87	90	93
Veg-Bld-Sky	73	84	89	87	91	93
All w/o (x,y)	64	78	85	83	88	91
Baselines	**5m**	**10m**	**20m**	**5m**	**10m**	**20m**
SIFT+(x,y)	84	89	91	90	92	93
SIFT	81	86	88	88	89	90
Toft [10]	60	73	80	78	85	88

We used two sequences captured on the west route, as shown in Figure 4 right. The two sequences were captured on 5 AM and 9 AM, which were under sunrise and daylight conditions, respectively. The sequence at 9AM contains more dynamic class objects than that at 5AM. We resized the images from their original size 1280×960 to 640×480 pixels. The images were captured at 15 fps while the GPS coordinates were measured at 10 FPS. Similar to SLC, we sampled the route captured on 9AM as the database of 3254 images and computing the VLAD codebook, and the route captured on 5AM for querying (2207 images).

V. EXPERIMENTS

A. Settings

CASENet: We use the CASENet model pre-trained on the Cityscapes dataset [54]. It contains 19 object classes that are also seen in our testing video sequences. We used nVidia Titan Xp GPUs to extract CASENet features, which can process around 1.25 images per second using CASENet original code. We did not retrain CASENet for our datasets, since getting ground truth semantic edges is a tedious manual task. We observed that the pre-trained model was sufficient to provide qualitatively accurate semantic edge features.

VLAD: We compared the CASENet-based semantic edge features to SIFT [2], and used VLAD to aggregate both to descriptors for image retrieval. To decide the number of clusters for VLAD, we find the optimal cluster numbers within 16, 32, 64 and 256 by experiments, with MiniBatchKMeans of at most 10,000 iterations. Our experiments showed that 64 clusters for CASENet features, 16 for SIFT on SLC Marathon dataset and 32 for SIFT on other datasets are the most optimal, and thus we applied these cluster numbers for further experiments. Note that although CASENet feature dimension is much smaller than SIFT (19 vs. 128), there are more CASENet features for each image as we get them for each pixel. As a result, CASENet works better with more clusters than SIFT. The VLAD of both were trained on CPUs. With Intel(R) Xeon(R) E5-2640 CPU and 125GB of usable memory, the training for 3000 images took about 30 minutes.

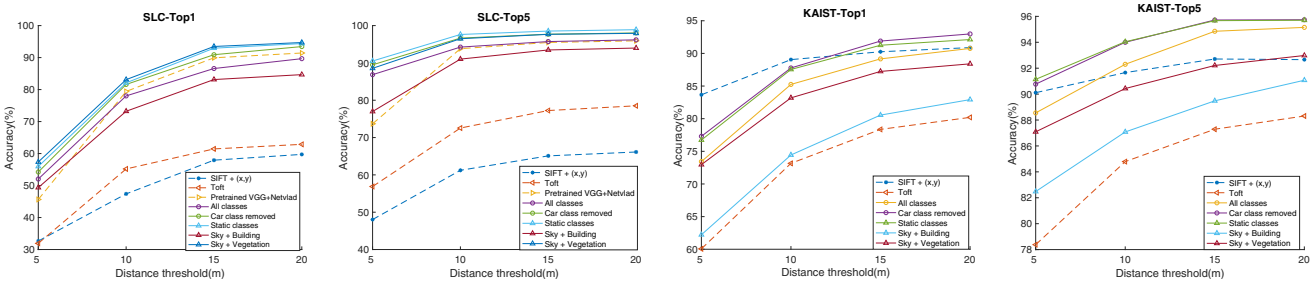
Evaluation criteria: We measured both top-1 and top-5 retrieval accuracy under different distance thresholds (5, 10, 15, and 20 meters). If any of these top-N retrieved images is within the distance threshold of the query image, we counted it as a successful localization.

B. Results and Ablation Studies

Figure 5 shows our main results compared with several baselines. Figure 8 presents several best and worst matching examples by our method. We also performed ablation studies on the importances of 1) object classes and 2) spatial coordinates used for feature augmentation, in Tables I and II.

Object classes: We first investigated the importance of different subsets of the 19 Cityscapes classes for localization (all augmented by 2D spatial coordinates) with two goals. The first is to evaluate individual class contributions to the accuracy. The second is to compare our approach with existing ones that also use semantic boundaries but with fewer classes. For example, one of the popular localization cues is skylines (edges between building and sky) [5]–[8].

For SLC and in most cases, removing dynamic classes (listed in the second half of the first block of Table I)



(a) (b) (c) (d)

Fig. 5. Localization accuracies. (a) and (b) represent the results for SLC Urban dataset while (c) and (d) represent the results for KAIST dataset. The x-axis represents the distance threshold and the y-axis represents the accuracy. Non-CASENet results are shown using dashed lines. No weighting of features are applied. Note for KAIST, the pretrained VGG-NetVLAD performances are very low (and even with retraining), thus we do not include them here. Note CASENet is not retrained either.

yields better accuracy than all classes, e.g., removing cars improves the accuracy by 2%. Note in some cases, removal of some dynamic classes causes minor drops in accuracy, e.g., removing motorcycle and bus, which we believe is insignificant, and mainly due to the lack of those classes in our dataset. As per our expectation, using only static classes (the 11 out of 19 classes) of CASENet performs better than using all classes, for both datasets. Specifically, building, sky and wall are the top 3 individual contributors, as removing them causes highest drop in the accuracy. Also using only vegetation, sky and building is comparable to using all static classes. Note that performances decrease for all methods in SLC Marathon. This dataset is more challenging due to a long portion of less discriminative suburb routes that contain fewer buildings than vegetation.

For KAIST, vegetation seems to be the most important individual class. Removing it causes the highest drop in the accuracy. Building and sky classes individually does not seem very significant. Again, using only static CASENet features performs better than any other feature combination.

Spatial coordinates: Besides object classes and their probabilities, we also tried removing the 2D-image-coordinate augmentation from the feature descriptors for both CASENet and SIFT. Surprisingly, this augmentation boosted the performance of both SIFT and CASENet by a large margin: SIFT+(x,y) vs. SIFT, and All vs. All w/o (x,y) in Table I and II. While this result seems counter-intuitive due to the loss of invariance in feature descriptors, the on-road vehicle localization is a more restricted setup and such constraints lead to high-accuracy localization.

A natural concern for such direct augmentation is the weighting of spatial coordinates compared with object class probabilities or SIFT features, which have larger dimensions. Thus we investigate the effect of a weighted feature augmentation as $\bar{\mathbf{Y}} = [\alpha\mathbf{Y}_1, \cdots, \alpha\mathbf{Y}_K, (1-\alpha)\mathbf{Y}_x, (1-\alpha)\mathbf{Y}_y]$, where $K = 19$ for CASENet and $K = 128$ for SIFT, $\mathbf{Y}_x, \mathbf{Y}_y$ indicate normalized 2D spatial coordinates. In Figure 6 and 7, we show that the combination of the two achieves the best performance, and higher weights should be given to spatial coordinates due to a smaller number of dimensions.

In summary, CASENet-VLAD generally performs better than SIFT-VLAD (and also augmented SIFT-VLAD for SLC), although the augmentation sometimes makes SIFT comparable to CASENet. For example, augmented SIFT features performed better than CASENet on KAIST, since without augmentation CASENet already performed worse than SIFT (Figure 5). We conjectured the main reason to be the different data distributions between the KAIST and Cityscapes, leading to degraded quality of CASENet features without domain adaption. Note that another deep baseline [10], pretrained on the Cityscapes, also performs worse than SIFT on KAIST.

Other deep baselines: We also compare with three deep baselines: 1) Toft et al.'s coarse localization method [10], which performs semantic segmentation using a pre-trained network [55] and computes a descriptor by combining histograms of static semantic classes as well as gradient histograms of building and vegetation masks in six different regions of the top half of the image; 2) VGG-NetVLAD [9]; and 3) PoseNet [12], a convolutional neural network that regresses the 6-DOF camera pose from a given RGB image. The results of the first deep baseline (our own implementation) and VGG-NetVLAD (the best pre-trained weights from the Pittsburgh dataset provided in [9]) are shown to be worse than CASENet in Figure 5. Note for KAIST, the pretrained VGG-NetVLAD performances are very low, and even with retraining the performance is still below 30%, thus we exclude them from Figure 5. For the application of PoseNet in this paper, instead of the 6-DOF output, we only regress 3 values from an image: the x-, y-location, and the orientation of the vehicle. Based on our initial experiments, we observed that the performance of PoseNet is less than 50%. This is much lower than other methods tested in this paper (Figure 5). We plan to investigate this further, but the high error could be due to the fact that the restricted pose parameters from the on-road vehicles (mostly straight lines and occasional turns) is insufficient to train the network.

VI. Discussion

We proposed a simple method to achieve high-accuracy localization using recently introduced semantic edge features [4]. While SIFT is one of the earliest feature descriptor used for localization, SIFT-VLAD is still considered as the state-of-the-art localization algorithm. We show significant

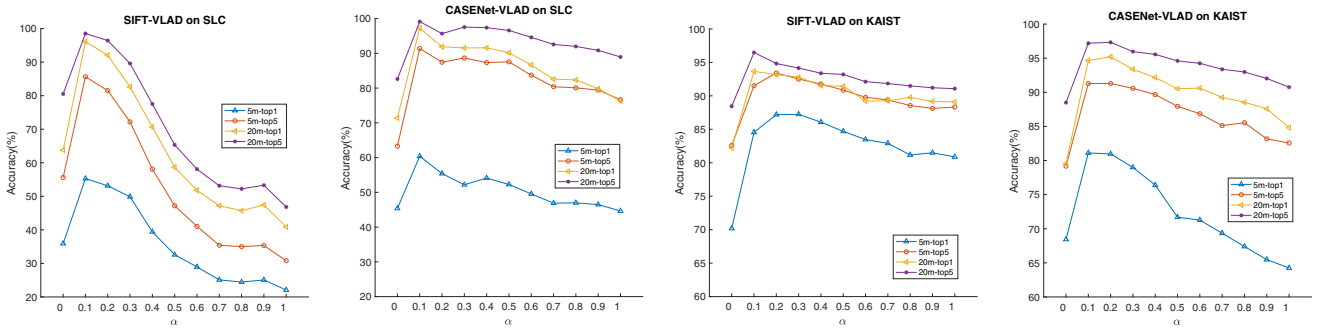


Fig. 6. Effect of weighted spatial coordinate augmentation on SLC Urban (left) and KAIST (right). At the optimal $\alpha = 0.1$, CASENet is still better than SIFT.

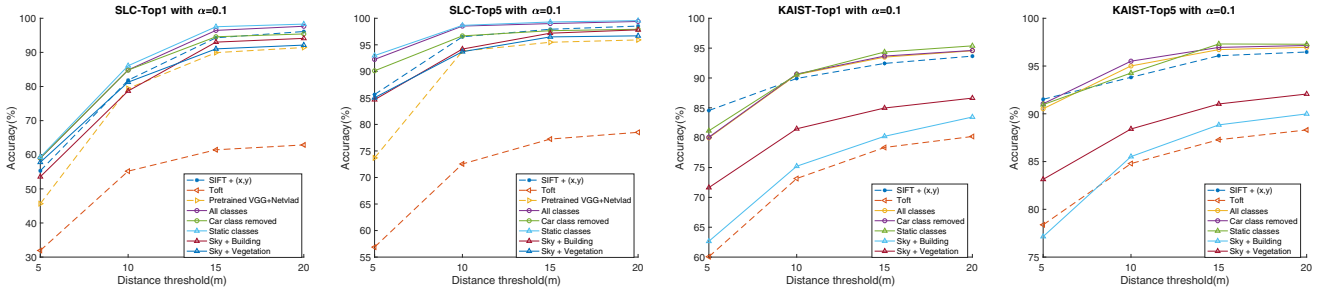


Fig. 7. Localization accuracies using weighted augmentation, with $\alpha = 0.1$ found to be optimal for both SIFT and CASENet (on SLC Urban and KAIST). Other settings are the same as in Figure 5. Note Toft [10] and NetVLAD are not weighted.

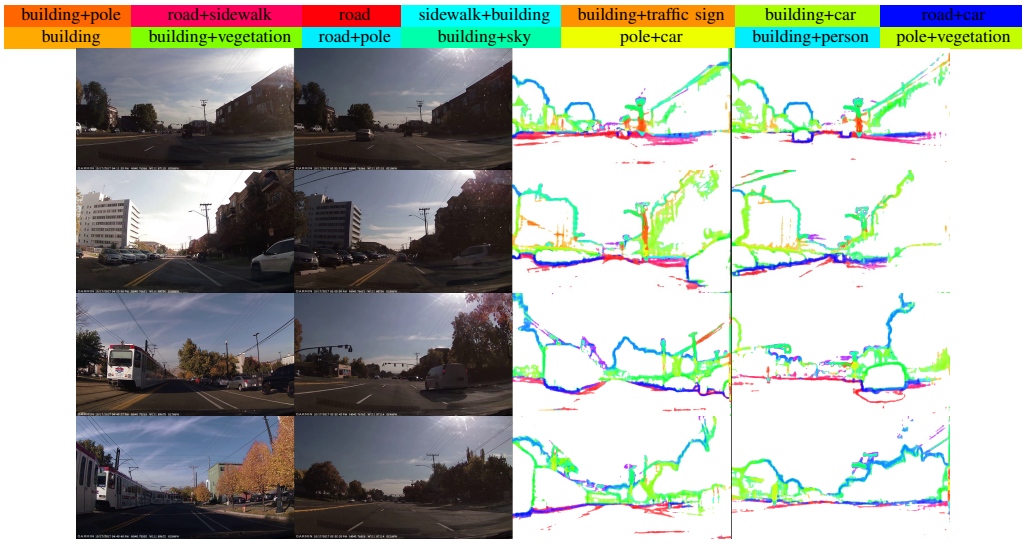


Fig. 8. Successful and failed matches of CASENet+VLAD. The top 2 rows show good matches . The bottom 2 rows show two of the worst results where the true distance is greater than 2 kms. In the 3rd row, the presence of dynamic object such as the train might lead to the high error.

improvement over the standard SIFT-VLAD, and the augmented SIFT-VLAD method. While the CASENet features are trained only on Cityscapes dataset, the pretrained model was sufficient for achieving state-of-the-art localization accuracy. Another interesting result that came out of our analysis is that skyline (either from building and sky, or from vegetation and sky) is a very powerful localization cue.

While the main localization idea is simple, we believe that this work unifies several ideas in the community. Furthermore, it has already been shown that semantic segmentation and depth estimation are closely related to each other [56], [57]. This paper takes a step towards showing that semantic segmentation and localization are also closely related, making one more argument towards holistic scene understanding. We will release the SLC datasets and code for research purposes.

ACKNOWLEDGMENTS

We specially thank the reviewers and area chairs for the feedback.

REFERENCES

[1] D. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, 2004. 1

[2] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," *PAMI*, vol. 34, no. 9, pp. 1704–1716, Sept. 2012. 1, 2, 5

[3] Y. Choi, N. Kim, K. Park, S. Hwang, J. Yoon, and I. Kweon, "All-day visual place recognition: Benchmark dataset and baseline," in *CVPR 2015 Workshop on Visual Place Recognition in Changing Environments*, 2015, pp. 8–10. 1, 2, 3, 4

[4] Z. Yu, C. Feng, M. Y. Liu, and S. Ramalingam, "Casenet: Deep category-aware semantic edge detection," *arXiv preprint arXiv:1705.09759*, 2017. 1, 3, 6

[5] M. Bansal and K. Daniilidis, "Geometric urban geo-localization," in *CVPR*, 2014. 1, 2, 5

[6] J. Meguro, T. Murata, H. Nishimura, Y. Amano, T. Hasizume, and J. Takiguchi, "Development of positioning technique using omni-directional ir camera and aerial survey data," in *Advanced Intelligent Mechatronics*, 2007. 1, 2, 5

[7] S. Ramalingam, S. Bouaziz, P. Sturm, and M. Brand, "Localization in urban canyons using omni-skylines," in *IROS*, 2010. 1, 2, 5

[8] O. Saurer, G. Baatz, K. Koeser, L. Ladicky, and M. Pollefeys, "Image based geo-localization in the alps," *IJCV*, 2015. 1, 2, 5

[9] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *CVPR*, 2016. 2, 3, 6

[10] C. Toft, C. Olsson, and F. Kahl, "Long-term 3d localization and pose from semantic labellings," in *ICCV workshop*, 2017. 2, 3, 5, 6, 7

[11] N. Piasco, D. Sidib, C. Demonceaux, and V. Gouet-Brunet, "A survey on visual-based localization: On the benefit of heterogeneous data," *Pattern Recognition*, 2018. 2

[12] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *ICCV*, 2015. 2, 6

[13] T. Weyand, I. Kostrikov, and J. Philbin, "Planet-photo geolocation with convolutional neural networks," in *ECCV*, 2016. 2

[14] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother, "Dsac-differentiable ransac for camera localization," in *CVPR*, 2017. 2

[15] T. Sattler, W. Maddern, A. Torii, J. Sivic, T. Pajdla, M. Pollefeys, and M. Okutomi, "Benchmarking 6dof urban visual localization in changing conditions," *arXiv preprint arXiv:1707.09092*, 2017. 2

[16] O. Koch and S. Teller, "Wide-area egomotion estimation from known 3d structure," in *CVPR*, 2007. 2

[17] F. Stein and G. Medioni, "Map-based localization using the panoramic horizon," in *IEEE Transactions on Robotics and Automation*, 1995. 2

[18] J. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *IROS*, 2008. 2

[19] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera pose voting for large-scale image-based localization," in *ICCV*, 2015. 2

[20] T. Sattler, B. Leibe, and L. Kobbelt, "Improving image-based localization by active correspondence search," in *ECCV*, 2012. 2

[21] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua, "Worldwide pose estimation using 3d point clouds," in *ECCV*, 2012. 2

[22] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi, "Visual place recognition with repetitive structures," in *CVPR*, 2013. 2

[23] A. Majdik, D. Verda, Y. Albers-Schoenberg, and D. Scaramuzza, "Air-ground matching: Appearance-based gps-denied urban localization of micro aerial vehicles," *J. Field Robot.*, 2015. 2

[24] S. Ramalingam, S. Bouaziz, and P. Sturm, "Pose estimation using both points and lines for geo-localization," in *ICRA*, 2011. 2

[25] B. Micusik and H. Wildenauer, "Descriptor free visual indoor localization with line segments," in *CVPR*, 2015. 2

[26] A. Armagan, M. Hirzer, P. M. Roth, and V. Lepetit, "Learning to align semantic segmentation and 2.5d maps for geolocalization," in *CVPR*, 2017. 2

[27] H. Badino, D. Huber, Y. Park, and T. Kanade, "Real-time topometric localization," in *ICRA*, 2012. 2

[28] M. A. Brubaker, A. Geiger, and R. Urtasun, "Lost! leveraging the crowd for probabilistic visual self-localization," in *CVPR*, 2013. 2

[29] N. Jacobs, S. Satkin, N. Roman, R. Speyer, and R. Pless, "Geolocating static cameras," in *ICCV*, 2007. 2

[30] D. Robertson and R. Cipolla, "An image-based system for urban navigation," in *BMVC*, 2004. 2

[31] W. Zhang and J. Kosecka, "Image based localization in urban environments," in *3DPVT*, 2006, pp. 33–40. 2

[32] M. Cummins and P. Newman, "Appearance-only slam at large scale with fab-map 2.0," *International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2011. 2

[33] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras," *CoRR*, vol. abs/1610.06475, 2016. 2

[34] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *CoRR*, vol. abs/1607.02565, 2016. 2

[35] J. Hays and A. Efros, "Im2gps: estimating geographic information from single images," in *CVPR*, 2008.

[36] E. Kalogerakis, O. Vesselova, J. Hays, A. Efros, and A. Hertzmann, "Image sequence geolocation with human travel priors," in *ICCV*, 2009. 2

[37] T. Lee, S. Patil, S. Ramalingam, Y. Taguchi, and B. Benes, "Barcode: Global binary patterns for fast visual inference," in *2017 International Conference on 3D Vision (3DV)*, 2017. 2

[38] M. J. Milford, G. Wyeth, and D. Prasser, "Ratslam: A hippocampal model for simultaneous localization and mapping," in *ICRA*, 2004. 2

[39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015. 2

[40] Y. Tian, C. Chen, and M. Shah, "Cross-view image matching for geo-localization in urban environments," *arXiv preprint arXiv:1703.07815*, 2017. 2

[41] N. N. Vo and J. Hays, "Localizing and orienting street views using overhead imagery," in *ECCV*, 2016. 2

[42] T. Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *CVPR*, 2015. 2

[43] S. Pillai and J. Leonard, "Self-supervised place recognition in mobile robots," in *Learning for Localization and Mapping Workshop, IROS*, 2017. 2

[44] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *NIPS*, 1994. 2

[45] F. Walch, C. Hazirbas, L. eal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-based localization with spatial lstms," *CoRR*, vol. abs/1611.07890, 2016. 2

[46] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *T-RO*, 2016. 2

[47] X. Sun, Y. Xie, P. Luo, and L. Wang, "A dataset for benchmarking image-based localization," in *CVPR*, 2017. 2

[48] E. Zemene, Y. Tariku, H. Idrees, A. Prati, M. Pelillo, and M. Shah, "Large-scale image geo-localization using dominant sets," *arXiv preprint arXiv:1702.01238*, 2017. 2

[49] P. Gronat, G. Obozinski, J. Sivic, and T. Pajdla, "Learning and calibrating per-location classifiers for visual place recognition," in *CVPR*, 2013. 2

[50] J. L. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler, "Semantic visual localization," *arXiv preprint arXiv:1712.05773*, 2017. 3

[51] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *CVPR*, 2006. 2

[52] G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," in *CVPR*, 2007. 2

[53] J. Lee, S. Lee, G. Zhang, J. Lim, and I. S. W.K. Chung, "Outdoor place recognition in urban environments using straight lines," in *ICRA*, 2014. 2

[54] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *CVPR*, 2016. 5

[55] G. Ghiasi and C. C. Fowlkes, "Laplacian pyramid reconstruction and refinement for semantic segmentation," in *ECCV*, 2016. 6

[56] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. Yuille, "Towards unified depth and semantic prediction from a single image," in *CVPR*, 2015. 7

[57] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *ICCV*, 2015. 7