

Find your Way by Observing the Sun and Other Semantic Cues

Wei-Chiu Ma¹ Shenlong Wang² Marcus A. Brubaker² Sanja Fidler² Raquel Urtasun²

¹Carnegie Mellon University ²University of Toronto

Abstract

In this paper we present a robust, efficient and affordable approach to self-localization which does not require neither GPS nor knowledge about the appearance of the world. Towards this goal, we utilize freely available cartographic maps and derive a probabilistic model that exploits semantic cues in the form of sun direction, presence of an intersection, road type, speed limit as well as the ego-car trajectory in order to produce very reliable localization results. Our experimental evaluation shows that our approach can localize much faster (in terms of driving time) with less computation and more robustly than competing approaches, which ignore semantic information.

1. Introduction

Self-localization is a crucial component required to make self-driving cars a reality. An autonomous system has to be able to drive from point A to point B, park itself and recharge its battery when needed. With the availability of maps, semantic scene understanding becomes easier when the car is localized as strong priors from the map can be exploited [38, 29]. Self-localization is also key for map building.

The most commonly used self-localization technique is the Global Positioning System (GPS), which exploits triangulation from different satellites to determine the position of the GPS device. However, low-cost GPS systems are not reliable enough for applications such as robotics or self-driving cars. The presence of skyscrapers, signal jammers and narrow streets are common sources of problems, that make these systems non robust.

To overcome the limitations of GPS, many *place recognition* techniques have been developed in the past few years. These approaches record how the “world” looks like either in terms of geometry (e.g., LIDAR point clouds) or visual features, and frame localization as a retrieval task, where one has to search for a similar place in the large-scale dataset of the world [1, 8, 18, 24, 25, 34, 35, 43]. This is typically combined with GPS, which narrows down the region of interest where the search needs to be performed. The

main limitation of place-recognition approaches is that they require an up-to-date representation of the whole world. This is far from trivial, as the world is constantly changing and in the case of visual features, one needs to capture over different seasons, weather conditions and possibly times of the day. Privacy is also an issue, as recording is currently illegal in countries such as Germany, where cameras can be used for driving but their content cannot be stored.

With these problems in mind, Brubaker et al. [4] developed an approach to self-localization that exploits freely available maps from OpenStreetMaps (OSM) and localizes based solely on visual odometry. The idea behind is that the vehicle’s trajectory is a very strong indicator of which roads the vehicle could potentially be driving on, and if one drives long enough, the car’s possible location can be narrowed down to a single mode in the map. This paradigm is much more appealing than place recognition approaches, as it does not require to know/store the appearance of the world, and only a cartographic map of the road topology is necessary.

Brubaker et al. [4] showed very impressive results in terms of localization accuracy, reaching the precision of the map. However, their approach suffers from three main problems. First, it can fail to localize in very dense road maps as it relies solely on the uniqueness of the ego-motion. Second, the time to localization remains fairly large, reducing its applicability. Last, the computational complexity is a function of the uncertainty in the map, which remains fairly large when dealing with maps that have repetitive structures (e.g., Manhattan grid).

In this paper, we push the limit of vision-based navigation systems. We propose to exploit semantics to reduce localization time and the computational cost, as well as to increase the number of sequences that can be successfully localized. We remain in the scenario where the appearance of the world is unknown in order to create affordable solutions to self-localization. Towards this goal, we develop a novel probabilistic localization approach which makes use of four different types of semantics in addition to the vehicle’s trajectory. The most important cue we exploit is the sun, which (if we know the time of the day) can act as a compass, providing information about the absolute orien-

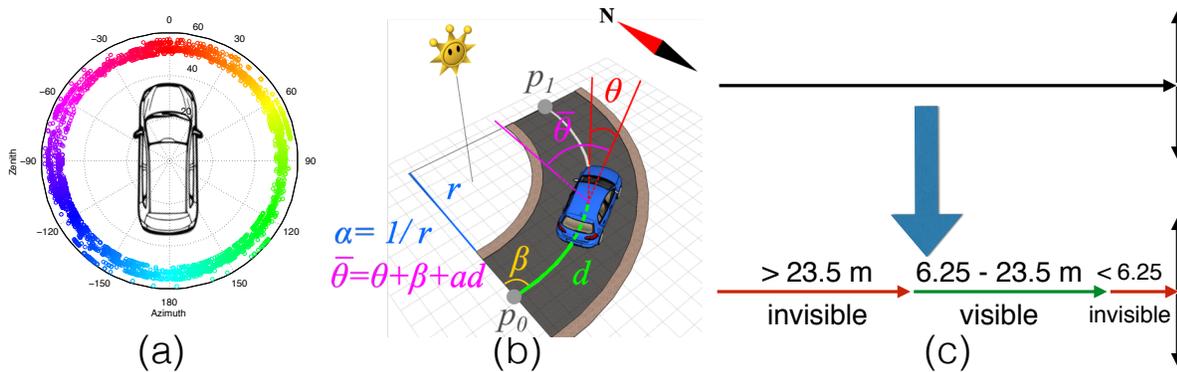


Figure 1: (a) The sun distribution in the KITTI Raw Dataset. (b) The parameterization used to describe a street segment as well as the position and the heading of a vehicle. (c) To encode intersection information, we partition road segments based on the visibility of an intersection.

tation of the vehicle. This provides very complementary information to visual odometry, which is rotation invariant as we do not know the initial orientation of the car. Additionally, we exploit the presence/absence of an intersection, the type of road that we are driving on, as well as the roads' speed limit. These cues can help us narrow down the set of possible car's locations in the map in a very short time, reducing the computational demand of our localization algorithm.

Estimating the sun direction is not an easy task, as the sun might not be present in the image. Traditional approaches [21] detect shadows and estimate the sun direction from them. Unfortunately, estimating shadows is far from trivial, and as a consequence existing approaches fail to provide accurate estimates of the sun direction. Instead, we take an alternative approach and employ a convolutional network to directly estimate the sun direction from a single image. As shown in our experiments, this simple approach works remarkably well in real-world driving scenarios. Interestingly, conv3 and conv4 activations fire at both shading and shadow regions. We also employ deep learning to estimate the presence/absence of an intersection and the road type. Importantly, we show that one can use OSM and the time of the day to create automatic labels for all the tasks (i.e., sun estimation, road type, presence/absence of an intersection).

We demonstrate the effectiveness of our approach on the challenging KITTI dataset [14] and show that we can localize much faster and with a lower computational cost than [4]. Furthermore, we successfully localize in scenarios where [4] fails. Next, we discuss related work, how to use deep learning to extract semantics as well as our novel self-localization approach. We then evaluate our approach and conclude.

2. Related Work

Localization in maps has been long studied in robotics, typically with particle-based Monte Carlo methods [6, 12, 17, 32]. These approaches mainly employ wheel odome-

try or depth measurements as observations. Maps have also been used to improve upon noisy GPS signals [3, 9, 11, 16]. In contrast, in this work we assume no knowledge of the initial position of the vehicle beyond a broad region of interest (which contains more than 2000km of road).

Place recognition methods attempt to perform self-localization without GPS, by searching for similar scenes in a large database of geo-tagged images [1, 18, 25, 34, 43, 23, 41], 3D point clouds [24, 30, 39, 2], 3D line segments [2] or driving trajectories [8]. These approaches typically have limitations as the database of images must be captured and kept up-to-date. Very recent work [31, 26] has started to make image-based localization invariant to certain appearance changes, however, they are still not very robust to severe changes in visual appearance due to weather and illumination.

Localization in road maps using visual cues in the form of the car's ego-trajectory was recently studied in [4, 10]. The advantage of this line of work over the retrieval-based approaches is that it only requires a cartographic map of the road network. Projects such as OpenStreetMap already provide world coverage and can be freely downloaded from the web. While [10] requires an initial estimate of position and was only tested in very small maps, [4] showed impressive localization results where the region of interest was the whole city. [4] relies solely on visual odometry, ignoring other visual and semantic cues which could be exploited for the localization task. In contrast, in this paper we propose to use the sun direction, the presence/absence of an intersection, the type of road we are driving on as well as the speed limit in addition to visual odometry as cues for localization. As shown in our experiments, we are able to localize faster, with less computation and with a higher success rate than [4].

Recent work exploited semantics for geo-localization. Shadows, direction of the sun [19, 42] and even rainbows [40] have been used for very rough localization (roughly 100km error) and camera calibration. These approaches require a long video recorded with a stationary

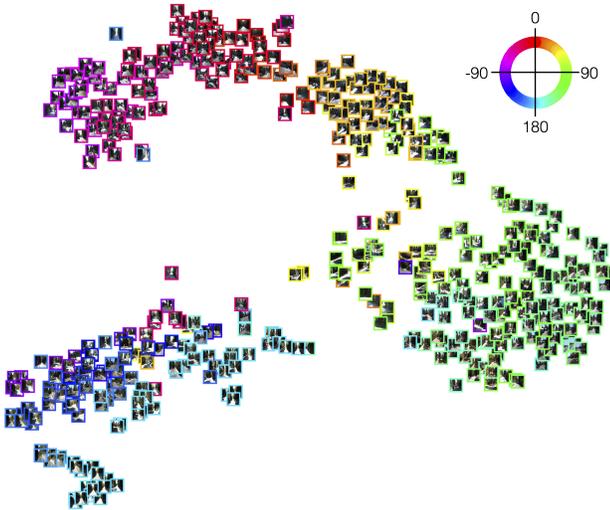


Figure 2: 2D embedding of the Sun-CNN feature space with t-SNE. Color of the image border denotes ground truth relative sun position, *i.e.* **red** sun is in front of the vehicle, **green** sun is on the right, **cyan** sun is on the back, and **magenta** sun is on the left. Sun-CNN not only effectively separates images showing different sun directions, but also preserves the *relative relationship*, *e.g.* images where sun are on the **front** lie between images where sun are on the **left front** and **right front**

camera, which is not realistic in autonomous driving scenarios. In [22], visual odometry was combined with sun and gravity sensors. However, additional sensors are required for this method, while our approach directly reasons about the sun direction from images. Another interesting work exploited semantic labeling from a single image and matching with the GIS dataset [5]. However, using semantics from a single image alone cannot achieve a meter-level accuracy in large-scale urban environments, where the semantic layout of the scene is very repetitive.

Semantic cues imposed by maps have also been used for indoor settings such as localization in apartments [27] and museums [28]. Their cues are tailored to static imagery, and thus not directly relevant for our scenario.

3. Deep Learning for Semantics

In this section we describe how we employ deep learning to estimate sun direction, road type as well as the presence/absence of an intersection.

3.1. Sun Direction Estimation

We start our discussion by describing how to estimate relative sun direction from a single image. Given the time of the day and a coarse geo-location, we can recover the absolute camera pose (*i.e.*, vehicle’s heading direction) by estimating the sun direction in the image, which can be used as a compass. As shown in the next section knowing the sun direction will significantly reduce localization time and

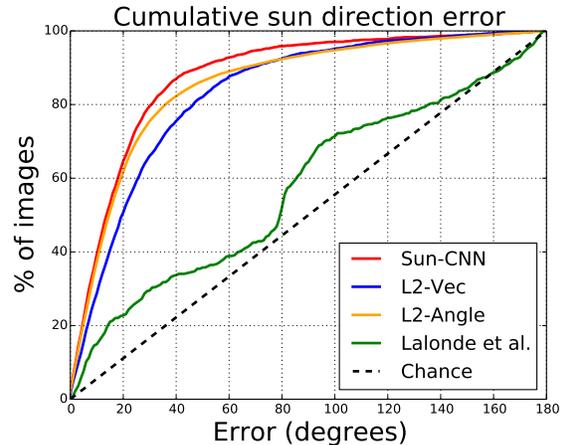


Figure 3: Cumulative sun direction error (angle between prediction and ground truth direction) for different methods on *KITTI-Sun* dataset. Our Sun-CNN outperforms previous state-of-the-art [21] and other parameterizations by a large margin.

computational cost.

Estimating illumination from a *single* RGB image is, however, an ill-posed problem due to the unknown depth/texture/albedo of the scene. Yet humans are good at estimating the light direction with the help of some visual cues *e.g.*, shadows, shading, over-exposed regions in the sky. For instance, if the buildings on the left side of the image are much brighter than the ones on the right side, we can infer that the sun is most likely on the right hand side. If we had an effective shadow detector and we knew the geometry of the scene, we could estimate the sun position. Unfortunately, neither shadow detectors nor shading estimation algorithms are good enough. Instead, in this paper we adopt a convolutional neural network to automatically learn and capture all kinds of visual cues that may help to estimate relative sun position.

In order to perform end-to-end training of the neural net we must have enough labeled data. However, to our knowledge no dataset with ground-truth labeled sun direction is large enough to train a deep neural network. Fortunately, given the timestamp and a coarse location (which we infer from GPS+IMU on KITTI [14]), we can calculate the absolute sun position in the sky under a horizontal coordinate system with the solar positioning algorithm [33]. We refer the readers to the appendix¹ for a detailed explanation of this algorithm. We thus automatically generate labels from the KITTI Raw dataset [14] to form our *KITTI-Sun* dataset. As adjacent frames are visually very similar, we subsample the videos at 1 frame/s, resulting in 3314 images. Note that the distribution of sun directions is quite uniform for our *KITTI-Sun* dataset as depicted in Fig. 1(a).

To ensure that our network takes into account the geodesics of the rotation manifold, we parameterize the sun

¹<http://www.cs.toronto.edu/~weichium/geo/appendix.pdf>

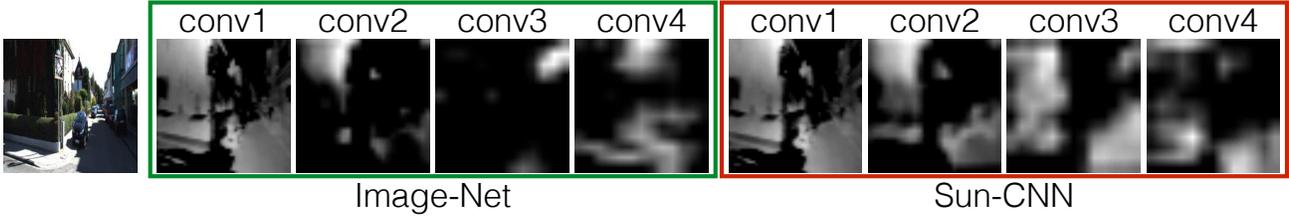


Figure 4: **Comparison between ImageNet-CNN and Sun-CNN:** the activation map of units at different layers for an input image. As the network goes deeper, the original ImageNet-CNN starts to capture certain high-level concepts, while our Sun-CNN focus on detecting the local illumination variation which is of crucial importance in detecting shadow and inferring sun position.

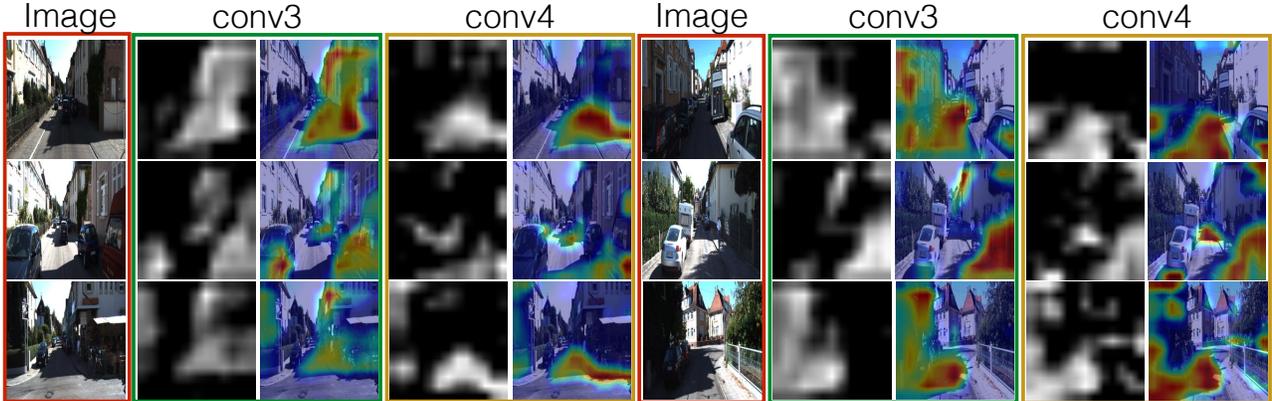


Figure 5: **Shading/shadow detectors emerge in Sun-CNN:** Test images and the corresponding activation maps of certain units in conv3 and conv4 layers of Sun-CNN. Despite being trained on *image-level* label (the relative sun position), our Sun-CNN automatically learns to fire on shadings (conv3) and shadows (conv4).

direction with a two dimension unit vector. Our network structure is adapted from AlexNet [20], where we replace the softmax layer with two continuous output variables representing the predicted 2D vector. By minimizing the distance between the ground truth vector and the output, the network is learning how to predict the sun direction correctly. We adopt cosine distance as our distance metric which measures the angle between the two vectors and in practice performs the best (see Sec. 5.1 for more details).

3.2. Intersection Classification

Unlike the sun direction, the presence of an intersection in an image cannot be directly estimated from GPS+IMU. An alternative is to use crowd-sourcing systems such as Amazon Mechanical Turk (MTurk). Labeling images, however, is an expensive process as a quality control process is frequently required in order to sanitize the annotations. Instead, we exploit GPS/IMU as well as map data (i.e., OSM) to automatically generate ground truth labels.

Similar to the *KITTI-Sun* dataset, we subsample the *KITTI-Raw* dataset [14] at 1 frame/s. We then locate the camera’s position and estimate the visible area in the map using the GPS+IMU information. We further exploit the fact that the field-of-view of *KITTI* is 135 degrees, and defined the intersection visible area to be a sector with ra-

dius (6.25m-23m). The radius is selected via an empirical in-house user study according to whether humans can reliably determine the presence of intersection. We then automatically label each image as containing an intersection if there is one in the sector of interest of the OSM map. Using this procedure, we obtained the *KITTI-Intersection* dataset, consisting of 3314 images, 518 of which contain an intersection. Note that although we focused on images from *KITTI*, our automatic labeling procedure can be generalized to other geo-tagged images. Our intersection classification network is adapted from GoogleLeNet [36].

3.3. Road Type Classification

We subsample the *KITTI-Raw* dataset [14] at 1 frame/s, and project the camera location of each image using GPS+IMU onto the nearest street segment on the OSM map. We then use the road type category provided by OSM to automatically create labels for the road-type classification task. We collapse labels $\{trunk, trunk-link, motorway, motorway-link\}$, into a single type *highway*, and all other labels into *non-highway*. Our *KITTI-road-type* dataset consists of 3314 images, 232 of which are non-highway. We adapt AlexNet [20] to deal with road-type classification.

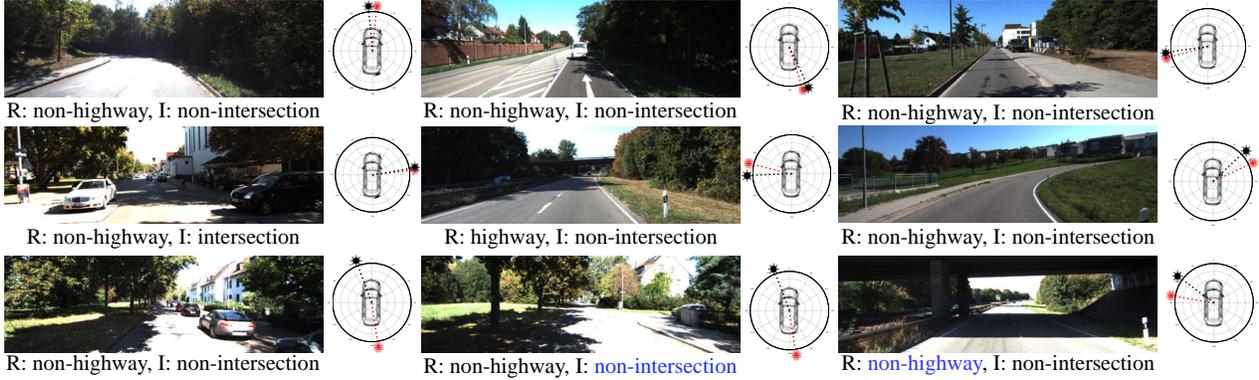


Figure 6: **Qualitative results of semantics estimated by our CNNs:** predictions of road and intersection types are under each image. The failure prediction is marked in blue. We also show predicted sun direction on the right of each image in black, while ground truth in red.

4. Sun and Semantics for Self-Localization

In this section we describe how to exploit OSM maps, sun direction, the presence/absence of an intersection, the road type, the speed limit and visual odometry to perform self-localization.

4.1. Map Representation

Following [4], we represent the map with a directed graph, where nodes encode street segments and edges define connections between the segments. As illustrated in Fig. 1(b), each street segment in the map is described by its starting and ending points \mathbf{p}_0 and \mathbf{p}_1 respectively, initial heading angle β , maximum speed limit V , road type R , intersection type I , and a curvature parameter α . The curvature is zero for linear street segments (*i.e.*, $\alpha = 0$) and $\alpha = \frac{\psi_1 - \psi_0}{\ell}$ for circular arc segments where ψ_0 and ψ_1 are the starting and ending angle of the arc and ℓ is the arc length of the segment. We further define the position and orientation of a vehicle in the map in terms of the street segment u that the vehicle is on, the distance traveled on that street segment d as well as the offset of the local street heading θ . Through this parametric representation, we can easily obtain the global heading of the vehicle $\bar{\theta}$, which is given by $\bar{\theta} = \theta + \beta + \alpha d$.

4.2. State-Space Model

The state of the model at time t is defined as $\mathbf{x}_t = (u_t, \mathbf{s}_t)$, where u_t is the street where the vehicle is driving on and $\mathbf{s}_t = (d_t, \hat{d}_{t-1}, \theta_t, \hat{\theta}_{t-1})$, with d_t and θ_t the distance and local heading angle of the vehicle w.r.t. the origin of the street u_t . Further, $\hat{d}_{t-1}, \hat{\theta}_{t-1}$ are the distance and angle at the previous time step $t - 1$ relative to current street u_t .

In this work, we employ five types of observations: sun direction, intersection type, road type, vehicle’s velocity, and visual odometry. Let $\mathbf{y}_t = (\phi_t, i_t, r_t, v_t, \mathbf{o}_t)$ be the observations at time t , with ϕ_t the estimated relative sun direction with respect to vehicle’s current heading, i_t the es-

timated intersection type in front of the vehicle, r_t the estimated type of road the vehicle is currently driving on, v_t the measured velocity of the vehicle (calculated from the visual odometry), and \mathbf{o}_t the visual odometry measurement. We assume that the observations are conditionally independent given the state \mathbf{x}_t , and define a fully factorized observation model:

$$p(\mathbf{y}_t | \mathbf{x}_t) = p(\phi_t | \mathbf{x}_t) p(i_t | \mathbf{x}_t) p(r_t | \mathbf{x}_t) p(v_t | \mathbf{x}_t) p(\mathbf{o}_t | \mathbf{x}_t) \quad (1)$$

We now describe each likelihood term in more details follow by the state transition distribution.

Sun Direction: This term encourages the estimate of the sun direction computed from the car’s location and time of the day to agree with the estimated sun direction with our Sun-CNN. Towards this goal, we model the sun direction as a Gaussian distribution:

$$p(\phi_t | \mathbf{x}_t) = \mathcal{N}(\phi_t | \mu_\phi(\mathbf{x}_t), \Sigma^s), \quad (2)$$

where $\mu_\phi(\mathbf{x}_t)$ is a deterministic function which computes the relative sun position from the vehicle’s current state \mathbf{x}_t and the global sun position that calculated from time of the day and coarse geo-location. Please refer to appendix for a detailed formulation of $\mu_\phi(\mathbf{x}_t)$. We learn the covariance Σ^s from the training set.

Intersection Type: This term captures the fact that the presence/absence of an intersection in a road segment should agree with the estimation of our Intersection-CNN. As mentioned in Sec. 3.2, we define an intersection to be visible if it lies between (6.25m-23m) and it intersects with the viewing frustum. Following this definition, we further partition the OSM street segments that contain an intersection into three sub-segments as shown in Fig. 1(c). The first one is connected to the intersection and has length 6.25m. As it is too close to the intersection, we assume that the intersection cannot be seen. The second segment spans from (6.25m-23m) and the intersection is visible. The third segment is

more than 23m away from the intersection and thus is labeled as not containing an intersection. Thanks to this partitioning, the intersection type does not vary within a street segment. We thus model this observation as a Bernoulli distribution only affecting the street segment ID:

$$p(i_t|\mathbf{x}_t) = p(i_t|u_t) = \gamma^{\delta_{i_t, u_t}^{int}} (1 - \gamma)^{1 - \delta_{i_t, u_t}^{int}}, \quad (3)$$

with δ_{i_t, u_t}^{int} a delta function with value 1 if the presence of an intersection in u_t is the same as the one output by the intersection-CNN, and 0 otherwise. We estimated γ using the confusing matrix of the training data. In practice we use $\gamma = 0.8$.

Road Type: This term captures the fact that roads of the same type as the one estimated by our Road-Type-CNN should have higher likelihood. We thus model it with a Bernoulli distribution:

$$p(r_t|\mathbf{x}_t) = p(r_t|u_t) = \beta^{\delta_{r_t, u_t}^{type}} (1 - \beta)^{1 - \delta_{r_t, u_t}^{type}}, \quad (4)$$

with δ_{r_t, u_t}^{type} a delta function with value 1 if the type of u_t is the same as the one output by the Road-CNN, and 0 otherwise. We estimated $\beta = 0.9$ using the confusing matrix of the training data.

Speed Limit: This term models the fact that the speed of driving depends on the speed-limit. We model the probability of the vehicle’s velocity as a piece-wise constant function, which depends on the road type. If the vehicle’s velocity is less than the maximum speed, we treat it as a uniform distribution over all possible speeds. We employ as maximum speed 25km/h over the speed-limit, as we empirically observed in KITTI that the driving speed is very often above the speed limit. A uniform distribution makes sense as there might be traffic ahead of us, slowing our driving. If the velocity exceeds the maximum speed, we give very low probability. Thus

$$p(v_t|\mathbf{x}_t) = p(v_t|u_t) = \begin{cases} \frac{0.99}{V_{u_t} + V_0}, & \text{if } v_t \leq V_{u_t} + V_0 \\ \epsilon, & \text{otherwise} \end{cases}, \quad (5)$$

where V_{u_t} denotes the maximum speed of street u_t and ϵ represents a very small number, which we set to 10^{-4} .

Odometry: Following [4] we model odometry as a Gaussian distribution linear in \mathbf{s}_t ,

$$p(\mathbf{o}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{o}_t|\mathbf{M}_{u_t}\mathbf{s}_t, \Sigma_{u_t}^o), \quad (6)$$

with $\mathbf{M}_{u_t} = [\mathbf{m}_d, \mathbf{m}_\theta]^T$, $\mathbf{m}_d = (1, -1, 0, 0)^T$ and $\mathbf{m}_\theta = (\alpha_u, -\alpha_u, 1, -1)^T$. Since the visual odometry performs significantly worse at higher speeds, we learn different variances for highways and city/rural roads as suggested by [4]. $\Sigma_{u_t}^o$ is therefore a function of the street u_t and is directly learned from data as discussed in Sec. 4.4.

Accuracy	Non-intersection	Intersection	Total
Intersection-CNN	82.8%	75.29%	81.62%
Human Perception	86.62%	79.53%	85.51%
Accuracy	Non-highway	Highway	Total
Road Type-CNN	99.45%	91.38 %	98.88%
Human Perception	99.51%	93.5 %	99.06%

Table 1: Accuracy for Intersection-CNN and Road-Type-CNN vs humans.

State Transition: We factorize the state transition distribution as

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = p(u_t|\mathbf{x}_{t-1})p(\mathbf{s}_t|u_t, \mathbf{x}_{t-1}).$$

As is common in the literature, we use a second order constant velocity model to describe the vehicles’ motion represented by linear transition dynamics corrupted with Gaussian noise for $p(\mathbf{s}_t|u_t, \mathbf{x}_{t-1})$. We define the probability of changing streets, $p(u_t|\mathbf{x}_{t-1})$, as a sigmoid, encoding the fact that transitions between street segments are more likely to occur as we arrive to the end of the street segment. We refer the readers to the appendix for more details.

4.3. Inference

Inference in our model consists of recursively computing the distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. The posterior can be factored as $p(\mathbf{x}_t|\mathbf{y}_{1:t}) = p(\mathbf{s}_t|u_t, \mathbf{y}_{1:t})p(u_t|\mathbf{y}_{1:t})$ using the product rule, where $p(u_t|\mathbf{y}_{1:t})$ is a discrete distribution over streets and $p(\mathbf{s}_t|u_t, \mathbf{y}_{1:t})$ is a continuous distribution over the position and orientation on a given street. The discrete distribution $p(u_t|\mathbf{y}_{1:t})$ is easily represented as a multinomial distribution over street labels. As for the continuous distribution $p(\mathbf{s}_t|u_t, \mathbf{y}_{1:t})$, we represent it with a Gaussian mixture model. With our model assumptions, the filtering distribution can then be written recursively as

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})}. \quad (7)$$

Here, we use the efficient approximation of [4] to approximate our posterior. This is possible as by design our likelihood terms are either Gaussian or only dependent on the street segment which is a discrete variable. More details on inference can be found in the appendix.

4.4. Learning

In this section, we discuss how we learn the odometry and sun direction noise, Σ_u^o and Σ^s . As both odometry and sun direction are modeled as a Gaussian distribution, we apply maximum likelihood estimation (MLE) to learn the variance from the data. For odometry, we compute the ground truth by projecting each image onto the road using GPS+IMU, and calculating the odometry of the GPS. For sun direction, we employ the images in the *KITTI-Sun* dataset. We note that different parameters were learned for

	Localization Time	Computational Time	Gini Index	Success Rate
Brubaker <i>et al.</i> [4]	46 ± 24s	0.69s/frame	0.765 ± 0.057	81.8%
Our Full Model	25 ± 21s	0.48s/frame	0.879 ± 0.013	90.9%

Table 2: **Comparison against [4]**. Our approach significantly outperforms [4] in terms of localization time, computation time, Gini index and success rate.

highways and city/rural roads as the visual odometry performs significantly worse at higher speeds. Leave-one-out cross validation was used to learn the parameters.

5. Experimental Evaluation

We validate the effectiveness of our approach on the training sequences from the KITTI visual odometry benchmark [14] where ground truth trajectory is available.

5.1. Deep Learning for semantics

For all three semantic tasks, we hold out images from each odometry sequence to create each split, and use the rest for training, resulting in 11 models. We augment the training images 10 times by cropping/mirroring. All hyperparameters are selected via grid search on the validation set. Note that sequence 03 is excluded from the semantics evaluation as GPS and IMU data is not available.

Sun-CNN: We initialized our network with weights pre-trained on ImageNet [7], and employ a learning rate of 10^{-5} for fc7, and 10^{-6} for the rest. Training a deep net for this task took around 3 hours requiring approximately 200 epochs to converge. We use the cumulative sun direction error (angle between prediction and ground truth direction) across the *KITTI-Sun* dataset as metric. We compare our model against the state-of-the-art sun estimation approach of Lalonde *et al.*[21]. We use default values for all parameters in [21], except the sun visibility v_s and the horizontal line v_h . The sun visibility probability $P(v_s)$ of all images are set to 0.8, since KITTI was collected in good weather. The horizontal line v_h is computed using KITTI calibration. For more details, we refer the readers to [21]. To verify our current settings best describe the geodesics of the rotation manifold, we also exploit different loss functions and parameterization. We first change the distance metric between two vectors from cosine distance to L2 distance, which we denoted as *L2-Vec*. We further re-parameterize the sun direction in the image with a single variable (angle) and modify the network structure to a single continuous output, which we denote as *L2-Angle*. Fig. 3 shows that our Sun-CNN significantly outperforms [21] by a large margin and our choice of loss function and parameterization consistently surpass those of others. Over 60% of the images have prediction errors less than 20 degrees with our approach, while only 25% for [21]. A few qualitative results are shown in Fig. 6. We also visualized the fc7 embedding space learned by Sun-CNN with t-SNE [37] in Fig. 2. Sun-

CNN not only separates images based on their sun direction, but also preserves the relationship among images, *i.e.*, images with similar sun direction are embedded closer.

To further understand why our Sun-CNN performs so well, we visualize in Fig. 4 the activation map of some units at different layers of the network as well as a network with same architecture but trained on the ImageNet classification task. Unlike classic network whose deep convolution layers learn high-level concepts [44], our kernels at conv3 and conv4 layers start to capture the illumination variations in the images, *e.g.* shadows and shadings. We argue that this is because our Sun-CNN learns that shading and shadows are crucial for predicting sun direction. As a proof of concept, Fig. 5 shows the activations of several neurons in conv3 and conv4 layers. Despite being trained on a *high-level* task (*i.e.*, sun direction), our Sun-CNN automatically learns to fire on both shadings and shadows.

Intersection-CNN: For each fold, we uniformly oversample the images labeled as intersections so that each class is balanced. Since we are trying to differentiate whether an image is taken at a *place* where an intersection is visible, we pre-trained the convolutional net on the MIT Places dataset [45]. We set the learning rate of the last fully connected layer to 6×10^{-4} , and the rest to 6×10^{-5} . Training took 2 hours and 150 epochs to converge. As shown in Tab. 1 our Intersection-CNN can achieve 82% accuracy. To validate human performance on this task, we also asked 3 in-house annotators to provide labels for each image. Tab. 1 shows that human perception is only 4% higher than our deep model, demonstrating the difficulty of this task. Some qualitative results are shown in Fig. 6.

Road-Type-CNN: As the data is unbalanced, we over-sampled the less frequent classes. We pre-trained our network on the MIT Places dataset [45]. The learning rate of fc7 is set to 10^{-2} , while we use 10^{-3} for the rest. On average, training took around 1.5 hours and approximately 100 epochs to converge. As shown in Tab. 1 the Road-Type-CNN can achieve almost 99% performance. We also validate human performance on this task by asking 3 in-house annotators to label each image. The performance of the Road-Type-CNN is comparable to human perception.

5.2. Localization

We subsample the KITTI sequences from 10 frame/s to 1 frame/s. Slower rates were found to suffer from excessive accumulated odometry error, while a higher rate may

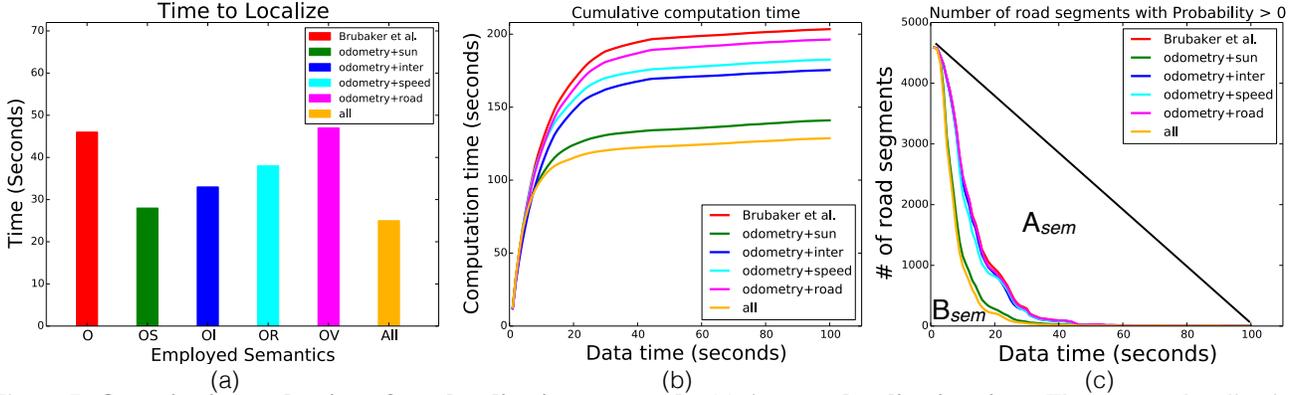


Figure 7: **Quantitative evaluation of our localization approach:** (a) **Average localization time.** The average localization time when employing different semantic cues. It is computed across the sequences that all methods can localize (b) **Cumulative computation time.** The average computation time when employing different semantic cues. We run our program on 8 cores with a basic Python implementation. Although our inference is slower than real-time at first, we argue that we can reduce it by employing more computational resource (*i.e.* using more cores). (c) **Road segments with probability larger than 0.** The average number of road segments (3m) with probability larger than 0 when employing different semantic cues. We denote the area below each curve as B_{sem} , while the area between the black line and the curve as A_{sem} , where sem denotes the semantic cues employed. In our scenario, the closer $\frac{A_{sem}}{A_{sem}+B_{sem}}$ to 1, the better. We compute the average using only the sequences that all methods localize.

suffer from correlations in semantic errors. We utilize LIBVISO2 [15] to compute visual odometry and velocity. For sun direction, presence of an intersection, and road-type, we employ the corresponding CNN models trained in the split that does not contain the test sequence. Note that sequences in KITTI have different routes, the training/test images are thus spatially non-overlapping as well. We tested our localization algorithm on both the full sized map of Karlsruhe, as well as the sequence-specific, cropped maps provided by [4]. The cropped maps include the region which contains the ground truth trajectory and, on average, cover an area of 2 km² and contain 100 km of drivable roads. While [4] pruned dirt roads and alleyways during preprocessing stage, we preserve all drivable roads in the map, making the localization problem more difficult (see appendix for comparison).

As shown in Tab. 2 our model using all semantic cues performs much better than [4] in terms of localization time, computation time and localization success rate ². A sequence is defined to be *localized* when, for at least ten seconds, there is a single mode in the posterior. Once the criteria for localization is met, all subsequent frames are considered localized. We note that we localize in sequences where [4] failed ³. To further evaluate how *fast* our approach can reduce the solution space, we also employ *Gini index* [13], a well-known measure of statistical dispersion, as a new met-

²We re-emphasize that the localization accuracy of [4] is already at level of precision of OSM and cannot be improved further. The accuracy of our approach is similar to [4]. Please refer to appendix for detailed numbers.

³One should note that as we employed a more complex map comparing to [4], the results of using merely visual odometry may differ from their paper.

ric. Fig. 7(b) shows the average number of road segments (3m) that have probability larger than 0 across all sequences when employing different semantic cues. We denote the area below each curve as B_{sem} , while the area between the black line and the curve as A_{sem} , where sem denotes the semantic cues employed. The Gini index is defined as $\frac{A_{sem}}{A_{sem}+B_{sem}}$. The closer the ratio is to 1, the faster the solution space is pruned (the better).

Examples of results on cropped maps are shown in Fig. 8 (see appendix for results using the full map). We observe that sequence 06 traverses a geometrically symmetric path, causing inference to be unable to select between the two modes using only visual odometry. When using the sun, this ambiguity can be resolved. Since the intersections along the path are distinct, intersection information can also resolve the symmetry. We also show the average localization time, the average computation time and how fast an approach can reduce the solution space across all sequences in Fig. 7. In general, additional semantics reduce the localization time and lower the computational cost by reducing uncertainty.

Tab. 3 shows more detailed ablation studies. The symbols “O”, “S”, “I”, “R”, “V” represent the observation types that were used during inference, *i.e.*, visual odometry, sun direction, intersection type, road type and velocity. Sequences that are not localized are indicated with a “*”. In general, adding each semantic cue will help boost the performance. However, if the semantic observations estimated from the images are noisy, they might lengthen the time required to localize. For instance, intersection-CNN hurts the localization in sequence 07 due to errors in estimating the sun direction, while velocity slightly increase localization time due to the over-speed of the vehicle. When combining

O	S	I	R	V	00	01	02	03	05	06	07	08	09	10	Average
✓					22s	90s	27s	36s	52s	*	27s	79s	30s	43s	46 ± 24s
✓	✓				20s	16s	22s	27s	43s	69s	13s	79s	26s	33s	28 ± 22s
✓		✓			22s	23s	25s	30s	51s	68s	34s	80s	18s	41s	33 ± 21s
✓			✓		23s	23s	27s	36s	52s	*	27s	79s	30s	42s	38 ± 18s
✓				✓	22s	90s	28s	34s	53s	*	27s	79s	36s	42s	47 ± 24s
✓	✓	✓	✓	✓	20s	12s	21s	21s	43s	31s	13s	80s	13s	15s	25 ± 21s

Table 3: **Quantitative evaluation:** "O", "S", "I", "R", "V" represent the observation types that were used during inference, i.e., visual odometry, sun direction, intersection type, road type and velocity. The average localization time, position and heading error are computed with sequences that localizes. Sequences that did not localize are indicated with a "*"*. No approach localize Sequence 04. Full table with all combinations of semantics and all metrics is shown in appendix. When employing all semantics our approach localizes faster.

all cues, our inference algorithm significantly improves the efficiency and achieves the best performance. One may suspect that the sun is not useful when cloudy. Note that our semantic cues are complementary and even without estimating the sun direction we still outperform [4] (see appendix). This is important in autonomous driving as one has to be robust.

To visualize how individual cues contribute to the localization task, we show frames from sequence 05 in Fig. 9. Sun direction provides the strongest initial cue, as it is able to fairly quickly eliminate a large number of roads whose directions are inconsistent with the heading implied by the sun. Speed limit information is most helpful when a vehicle is traveling at high speeds, enabling many low-speed roads to be pruned. However, even when traveling at slower speeds, it can be helpful as slower speeds are less likely on highways. Thus the portion of highway on the left has lower likelihood with speed information than with odometry alone. Intersection cues also help pruning the space. For instance, at $t = 26s$ the vehicle is not near an intersection and thus one can observe that modes near intersections are suppressed. Finally, the road type classifier by its nature is only useful in differentiating highways from other roads. For instance, at $t = 8s$ the highway has been pruned.

While the added cues make localization more robust and efficient, there are still limitations, particularly for extremely short or ambiguous sequences. Fig. 10 shows a sequence which still fails to localize, even with the added cues. We note that when compared to the odometry only case (top) the uncertainty has been further reduced (bottom), suggesting that with just a few more seconds of driving we would be able to localize by pruning out the faint secondary mode. In contrast, with only odometry, there still remains the ambiguity of the overall direction of travel.

6. Conclusions and Future Work

We have presented an effective affordable approach to self-localization, which exploits freely available maps as

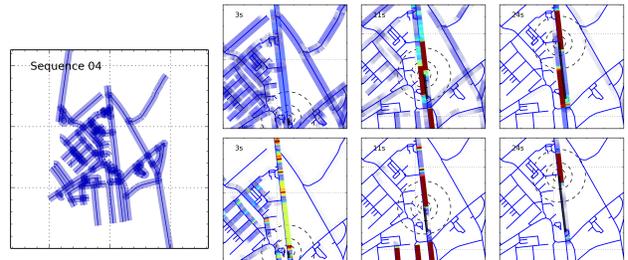


Figure 10: **Failure case:** The first row shows the results of [4], while the second row shows ours with all semantics. Although our model still fails to localize, we have reduced considerably the uncertainty. Just a few more seconds of driving, we would be able to localize by pruning the faint secondary mode using the intersection information.

well as visual odometry and semantic cues such as the sun direction, presence/absence of an intersection, road type and speed limits. Towards this goal, we exploited deep learning to directly estimate the semantics from monocular images. Interestingly, our Sun-CNN automatically learns coarse shading/shadow detectors. We have also shown how to automatically generate high quality labels without human intervention for all semantic tasks. We demonstrated the effectiveness of our localization approach in the challenging KITTI dataset and showed that we can localize faster and using less computation than [4]. In the future, we plan to exploit more semantic cues like traffic signs, trees, buildings, and road width. Although these data are either sparsely annotated or mostly incorrect, we hope the OSM maps become more complete and noise-free.

References

- [1] G. Baatz, K. Köser, D. Chen, R. Grzeszczuk, and M. Pollefeys. Leveraging 3D City Models for Rotation Invariant Place-of-Interest Recognition. *IJCV*, 2012.
- [2] M. Bansal and K. Daniilidis. Geometric urban geo-localization. In *CVPR*, 2014.
- [3] S. Bonnabel and E. Salaün. Design and prototyping of a low-cost vehicle localization system with guaranteed conver-

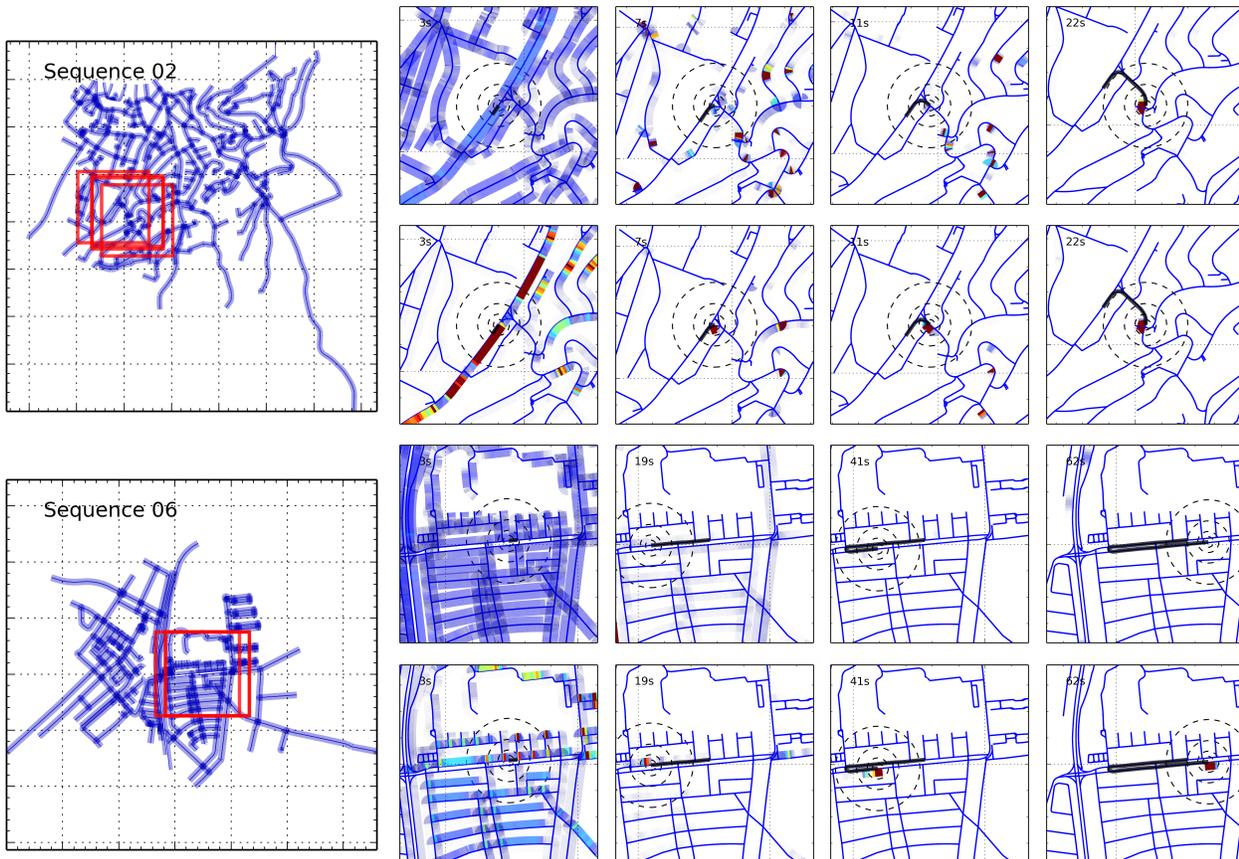


Figure 8: **Qualitative localization results compared to [4]:** Results of other sequences can be found in appendix. The left most column shows the full map region for each sequence, followed by zoomed in sections of the map showing the posterior distribution over time. The upper row is the result of [4], and the lower row is ours. The black line is the GPS trajectory and the concentric circles indicate the current GPS position. Grid lines are every 500m. High probability is indicated with red while low probability regions are shown in blue.

- gence properties. *Control Engineering Practice*, 19(6):591–601, 2011.
- [4] M. Brubaker, A. Geiger, and R. Urtasun. Lost! leveraging the crowd for probabilistic visual self-localization. In *CVPR*, pages 3057–3064. IEEE, 2013.
- [5] F. Castaldo, A. R. Zamir, R. Angst, F. Palmieri, and S. Savarese. Semantic cross-view matching. In *Vision from Satellite to Street - The Second International Workshop in Conjunction with ICCV 2015*, 2015.
- [6] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. *CVPR*, 1999.
- [7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE, 2009.
- [8] R. Dewri, P. Annadata, W. Eltarjaman, and R. Thurimella. Inferring trip destinations from driving habits data. In *WPES*, 2013.
- [9] M. E. El Najar and P. Bonnifait. A road-matching method for precise vehicle localization using belief theory and kalman filtering. *Autonomous Robots*, 19(2):173–191, 2005.
- [10] G. Floros, B. van der Zander, and B. Leibe. OpenStreet-SLAM: Global vehicle localization using OpenStreetMaps. In *ICRA*, 2013.
- [11] C. Fouque and P. Bonnifait. Matching raw gps measurements on a navigable map without computing a global position. *IEEE Trans. on Intelligent Transportation Systems*, 13(2):887–898, 2012.
- [12] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *AAAI*, 1999.
- [13] J. L. Gastwirth. The estimation of the lorenz curve and gini index. *The Review of Economics and Statistics*, pages 306–316, 1972.
- [14] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.
- [15] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium*, pages 963–968. IEEE, 2011.
- [16] J. Guivant and R. Katz. Global urban localization based on road maps. In *IROS*, pages 1079–1084, 2007.

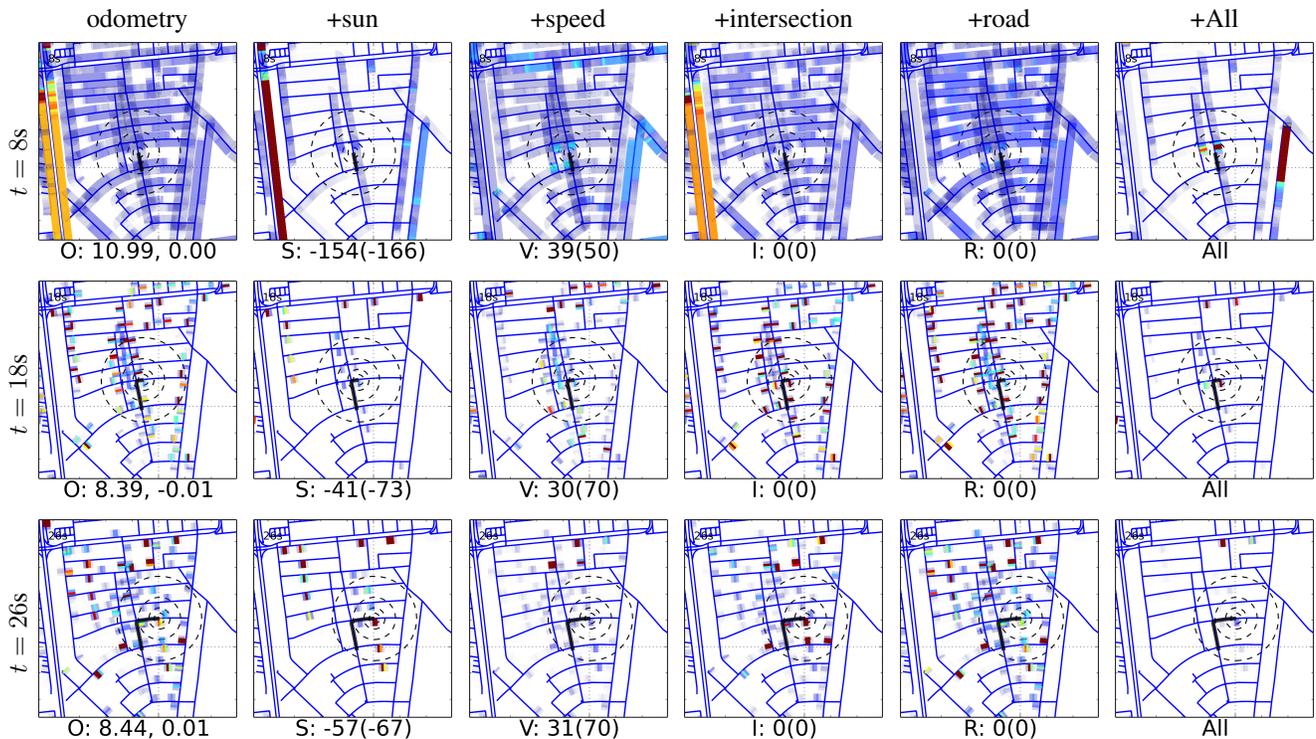


Figure 9: **Contribution of each semantics:** Results using different semantics at several time steps. The different semantics help reduce the uncertainty in the map. The observations and the ground truth (in parenthesis) are shown below each image.

- [17] J.-S. Gutmann, W. Burgard, D. Fox, and K. Konolige. An experimental comparison of localization methods. In *ICIRS*, 1998.
- [18] J. Hays and A. A. Efros. im2gps: estimating geographic information from a single image. In *CVPR*, 2008.
- [19] I. N. Junejo and H. Foroosh. Estimating geo-temporal location of stationary cameras using shadow trajectories. In *Computer Vision—ECCV 2008*, pages 318–331. Springer, 2008.
- [20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.
- [21] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan. Estimating natural illumination from a single outdoor image. In *ICCV*, 2009.
- [22] A. Lambert, P. Furgale, T. Barfoot, and J. Enright. Field testing of visual odometry aided by a sun sensor and inclinometer. *Journal of Field Robotics*, 29:426 – 444, 2012.
- [23] F. Li and J. Kosecka. Probabilistic location recognition using reduced feature set. In *ICRA*, 2006.
- [24] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *ECCV*, 2012.
- [25] T.-Y. Lin, Y. Cui, S. Belongie, J. Hays, and C. Tech. Learning deep representations for ground-to-aerial geolocation. In *CVPR*, pages 5007–5015, 2015.
- [26] C. Linegar, W. Churchill, and P. Newman. Work smart, not hard: Recalling relevant experiences for vast-scale but time-constrained localisation. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 90–97. IEEE, 2015.
- [27] C. Liu, A. Schwing, K. Kundu, R. Urtasun, and S. Fidler. Rent3d: Floor-plan priors for monocular layout estimation. In *CVPR*, 2015.
- [28] R. Martin-Brualla, Y. He, B. C. Russell, and S. M. Seitz. The 3D Jigsaw Puzzle: Mapping Large Indoor Spaces. In *ECCV*, 2014.
- [29] K. Matzen and N. Snavely. Nyc3dcars: A dataset of 3d vehicles in geographic context. In *ICCV*, 2013.
- [30] F. Moosmann and C. Stiller. Joint self-localization and tracking of generic objects in 3d range data. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1146–1152. IEEE, 2013.
- [31] P. Nelson, W. Churchill, I. Posner, and P. Newman. From dusk till dawn: Localisation at night using artificial light sources. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 5245–5252. IEEE, 2015.
- [32] S. M. Oh, S. Tariq, B. N. Walker, and F. Dellaert. Map-based priors for localization. In *ICIRS*, 2004.
- [33] I. Reda and A. Andreas. Solar position algorithm for solar radiation applications. *Solar energy*, 76(5):577–589, 2004.
- [34] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *ICCV*, 2011.
- [35] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *CVPR*, 2007.
- [36] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

- [37] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008.
- [38] S. Wang, S. Fidler, and R. Urtasun. Holistic 3d scene understanding from a single geo-tagged image. In *CVPR*, 2015.
- [39] R. W. Wolcott and R. M. Eustice. Visual localization within lidar maps for automated urban driving. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 176–183. IEEE, 2014.
- [40] S. Workman, R. P. Mihail, and N. Jacobs. A pot of gold: Rainbows as a calibration cue. In *ECCV*, pages 820–835. Springer, 2014.
- [41] S. Workman, R. Souvenir, and N. Jacobs. Wide-area image geolocalization with aerial reference imagery. *arXiv preprint arXiv:1510.03743*, 2015.
- [42] L. Wu, X. Cao, and H. Foroosh. Camera calibration and geolocation estimation from two shadow trajectories. *Computer Vision and Image Understanding*, 114(8):915–927, 2010.
- [43] W. Zhang and J. Kosecka. Image based localization in urban environments. In *3DPVT*, 2006.
- [44] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. In *ICLR*, 2015.
- [45] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, pages 487–495, 2014.