Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

Get to the Point: Learning Lidar Place Recognition and Metric Localisation Using Overhead Imagery

Tim Y. Tang, Daniele De Martini, and Paul Newman
Mobile Robotics Group, Oxford Robotics Institute, University of Oxford
Email: {ttang, daniele, pnewman}@robots.ox.ac.uk

***Abstract*—This paper is about localising a robot in overhead images using lidar. Specifically, we show how to solve both place recognition *and* metric localisation of a lidar using only publicly available overhead imagery as a map proxy. This is in contrast to current approaches that rely on prior sensor maps. To handle the drastic modality difference (overhead image vs. on the ground lidar), our method learns a representation that purposely and suitably transforms a given overhead image into a collection of 2D points, allowing for direct comparison against lidar scans. After both modalities are expressed as points, point-based methods can then be leveraged to learn the registration and place recognition task. Our method is the first to learn the place recognition of a lidar using only overhead imagery, and outperforms prior work for metric localisation with large initial pose offsets.**

I. Introduction

Localisation is a central task for autonomous navigation in large-scale, outdoor environments, and has been a core problem for the mobile robotics community for over two decades. For the purpose of this paper, we consider two types of localisation: topological and metric. Topological localisation, also known as place recognition, seeks to induce a rough estimate of the robot's pose, often apropos nothing. Given a coarse initial pose estimate from place recognition, metric localisation aims to compute a refined metric pose by registering live sensor data against some sort of prior map.

Lidar has always been a popular sensor for localisation. Both place recognition [7, 16, 23, 19] and metric localisation [48, 5, 2, 50] using lidar are well-studied problems, as well as methods that target global localisation [24, 12, 51, 38] by combining place recognition and metric localisation and outputting a metric pose, solving the "kidnapped robot" problem end-to-end. Existing methods rely on the proposition that a reliable, up-to-date lidar map is available, which may not be the case. In this paper we present an alternative approach to localise a lidar using only off-the-shelf, easily accessible overhead imagery, which often captures geometric entities also observable by ground lidar scans, providing cues for localisation.

An immediate challenge arises from the significant modality difference between overhead imagery and ground range sensors. To address the modality difference between satellite imagery and radar, [39] proposes to generate a synthetic radar image from a pair of satellite and radar images. The synthetic image is then in the same domain as live radar images and can be used for registration. The experimental results using the method in [39] are demonstrated for lidar metric localisation against overhead imagery in a follow-up work [40].

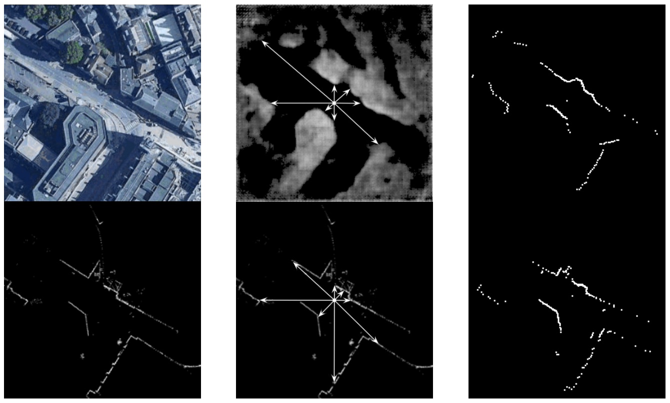
Fig. 1: Top row: given an overhead image (left), we learn an occupancy image (centre) that indicates how likely each pixel belongs to a space that will produce a point return if scanned by a lidar. This results in high probability for buildings and low probabilities for streets and roads. Ray-tracing the occupancy image from near the centroid results in a pseudo point-cloud (right). Bottom row: a lidar scan taken nearby, rotation-aligned with the overhead image for better visualisation (left). Ray-tracing the lidar image and keeping only the first return for each azimuth (centre), we arrive at a point-cloud (right) that can be directly compared against the pseudo point-cloud (top right).

The method in [39] has two major limitations. Firstly, it is strictly designed for metric localisation and as such cannot be directly applied to place recognition. Secondly, the quality of image generation in [39] is extremely dependent on the initial offset, resulting in pose errors increasing disproportionally with larger initial heading offsets. While the method in [39, 40] addresses the modality difference by generating synthetic lidar images from overhead imagery, we take a different approach where both overhead images and lidar scans are converted into a collection of 2D points, allowing for the use of standard point-based place recognition and pose estimation networks.

Specifically, given an overhead image, we learn an occupancy image that indicates how likely each pixel is to induce a range return by a ground lidar. This results in high probabilities for pixels on buildings and structures, and low probabilities for pixels on "free-space" such as roads and side-walks. We then ray-trace from near the centroid of the occupancy image along each azimuth until the first "occupied" pixel, emulating the resulting point-cloud if a lidar near the centroid takes a scan of its surrounding environment. This process is depicted in Figure 1 with an actual lidar scan taken at roughly the same position and rotation-aligned for visual reference.

To the best of our knowledge, our method is the first to learn the place recognition of a ground range sensor using only publicly available overhead imagery, and no GPS or prior sensor maps. We evaluate our method for both place recognition and metric localisation on publicly available datasets, and show that we outperform the prior work in [39] in metric localisation when the initial heading offset is large. Finally, we demonstrate that although designed for lidar, our method can also be used for radar place recognition and metric localisation against overhead imagery after appropriate domain transfer.

II. Related Work

A. Range Sensor Localisation against Overhead Imagery

While many works were proposed for visual localisation using overhead imagery [27, 29, 35, 31, 22, 11], localising a ground range sensor against overhead imagery remains a less-targeted problem due to the challenging modality difference.

Similar to dense image registration, [13] directly aligns lidar intensity maps onto satellite imagery using Normalised Mutual Information, and relies on the accumulation of several scans from accurate odometry. The methods in [25, 14] pre-process the overhead images by performing edge detection and semantic segmentation respectively, before comparing against range sensor data. Hussein et al. [20] localised a vehicle in a forest by matching tree stems detected by an on-board lidar against tree crowns observed in overhead imagery. Carle and Barfoot [9] localised a robot equipped with a lidar within orbital elevation maps by detecting features from topographic peaks. These methods are all examples of hand-crafted methodologies designed for specific setups, and may not necessarily generalise to more complex scenarios.

In light of this, RSL-Net [39] was proposed to use an end-to-end learning-based strategy for solving the metric localisation between satellite imagery and a ground radar or lidar, remaining free of hand-crafted features or methods. However, the work in [39] remains strictly limited to metric localisation, while our method aims to also handle place recognition.

B. Localisation Using Other Publicly Available Data

Other publicly available resources have also been shown to be useful for vehicle localisation, in particular road and building information from OpenStreetMap (OSM). The methods in [8, 17] localise by matching odometry trajectories to road paths from OSM. Panphattarasap and Calway [30] learned a light-weight descriptor to recognise intersections and gaps, and localised by comparing against descriptors of the operating environment built from building information in OSM. Yan et al. [49] utilised a similar strategy for localisation using OSM, but relied only on existing networks trained for point-cloud semantic segmentation, and eliminated the need to train new networks for recognising intersections and gaps.

Our method differs from [8, 17] in that we do not need a sequence of on-board measurements to compute odometry before a solution can be found. Compared to [30, 49], our method seeks to infer the *geometric* rather than *semantic* relationship between on-board sensor measurements and publicly available data, and can therefore perform geometric registration.

C. Learning-based Point-Cloud Registration

Learning-based methods for point-cloud registration have shown to be less prone to converging to local minima than classical approaches such as ICP [6]. The method in [53] learns local RGB-D descriptors with 3D convolutions for matching. 3DFeatNet [50] samples point-clusters using PointNet++ [33], and learns local descriptors using a triplet loss on point-clusters for feature alignment. Given a set feature descriptors, the method in [10] learns weights for point correspondences for downstream pose estimation, acting similar in spirit as an outlier rejection step. DCP [44] learns a per-point descriptor using DGCNN [45], finds point correspondences by matching learned descriptors, and computes the pose offset using singular value decomposition (SVD). In particular, the descriptors in [44] are optimised using only a loss on pose; the end-to-end differentiability is made possible by taking softmax rather than hard max when finding point correspondences. PRNet [43] builds upon DCP, allowing for partial-to-partial registration, by using Gumbel-Softmax for matching and iterative alignment during registration. DeepGMR [52] learns point-to-distribution rather than point-to-point correspondences by representing a point-cloud as a Gaussian Mixture during descriptor learning, and reduces computational complexity.

We utilise DCP [44] for registration and learning descriptors, as it outperforms other methods for our problem.

D. Deep Learning for Point-Cloud Place Recognition

A number of learning-based methods were proposed for large-scale point-cloud place recognition. PointNetVLAD [41] combines PointNet [32] and NetVLAD [1] to learn a per-point-cloud global descriptor for retrieval. LPD-Net [28] aggregates learned local features to produce a global descriptor for place recognition, and is shown to outperform [41]. PCAN [54] utilises attention to predict the significance of each local point feature, and favours task-relevant features when aggregating local features into a global one for retrieval. DH3D [15] learns local descriptors with a description loss using ground truth point correspondences, which can be used for registration. The local descriptors are then fed to a PointNetVLAD layer with attention to learn a global descriptor for place recognition. Barnes and Posner [3] learned key-points from radar images for odometry, and pooled local descriptors across spatial dimensions into a global one per image for place recognition.

Our approach to learning place recognition is similar as [15] in that we also re-use local descriptors learned for registration when learning the global descriptor. However, rather than applying a description loss, the local descriptors in our case are learned using DCP [44] with a loss on pose.

III. Overview

A. Problem Overview

We consider the problem of localising a ground lidar using overhead imagery. Specifically, we utilise satellite images from Google Maps [1], and project 3D lidar point-clouds to the $x-y$

[1] https://developers.google.com/maps/documentation/maps-static/overview

plane from a top-down perspective, producing 2D, "birds-eye" view lidar images. We discard points with z values less than 0 when creating the lidar images, to remove points on the ground. We denote satellite images and lidar images as I^S and I^L, respectively. Furthermore, all overhead images and lidar images are pre-processed to have the same resolution.

For place recognition, we consider a set of satellite images queried along known routes within an operating environment, $\{\mathrm{I}_i^S\}, i = 1, 2, \cdots, N_s$, and lidar images taken across time-stamps during live traversal $\{\mathrm{I}_j^L\}, j = 1, 2, \cdots, N_L$. For each I_j^L, we seek to find its nearest spatial neighbour from $\{\mathrm{I}_i^S\}$, thus providing a coarse estimate of the vehicle's global position as the latitude and longitude coordinates of each I_i^S are known. For simplicity, we assume the satellite images are queried along the same routes as the vehicle's live traversal, and that $N_S = N_L$, such that for every I_j^L, there is at least one correct spatial match from the set $\{\mathrm{I}_i^S\}$, and vice versa. For metric localisation, instead, given spatially proximal I_i^S in $\{\mathrm{I}_i^S\}$ and I_j^L in $\{\mathrm{I}_j^L\}$ with an unknown initial pose offset, we seek to solve their $SE(2)$ pose difference expressed as $\begin{bmatrix} x & y & \theta \end{bmatrix}$. For simplicity, from now on the indices i and j will be dropped in the context of metric localisation.

B. Image Generation vs. Point Learning

A widely used strategy for dealing with modality difference between two types of images is to apply image-to-image transfer methods such as Pix2Pix [21] or CycleGAN [55]. However, standard image-to-image transfer methods are not appropriate for transferring between satellite images and lidar images. Due to occlusion and a range sensor's limited field-of-view (FOV) compared to overhead images, regions in a satellite image may contain lidar returns in one satellite-lidar pair, but none in another, as shown in Figure 2. As a result, the mapping from I^S to I^L is one-to-many, which is not a function, and such a direct mapping is not suitable for learning with neural networks which are function approximators.

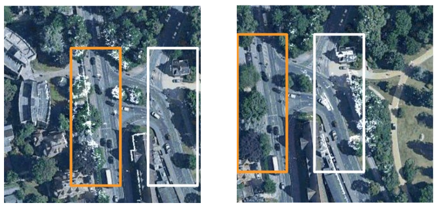
Fig. 2: Two satellite-lidar image pairs taken 10 seconds part, where lidar measurements (white points) are overlaid on top of satellite images with ground truth pose. The areas enclosed by the white and orange rectangles are identical patches in the satellite images. Image patches enclosed by the orange rectangles have lidar returns in the first satellite image, but none in the second; the reverse is true for image patches enclosed by the white rectangles.

To handle such ambiguity, the image generation in RSL-Net [39] is conditional on both a satellite image and a live radar image, where the exact appearance of the synthetic image is dictated by the live radar image, and the synthetic image is pixel-wise aligned to the satellite image. In particular, as CNNs are non-equivariant[2] to rotation [26], RSL-Net seeks to infer the rotation offset prior to image generation:

$$h(\mathrm{I}^L, \mathrm{I}^S) \rightarrow \mathrm{I}_*^L, \quad g(\mathrm{I}_*^L, \mathrm{I}^S) \rightarrow \hat{\mathrm{I}}^L, \tag{1}$$

where I^L and I^S are a pair of lidar and satellite images that are spatially proximal but have an unknown $SE(2)$ offset, I_*^L is I^L rotated to be rotation-aligned with I^S, and $\hat{\mathrm{I}}^L$ is a synthetic lidar image pixel-wise aligned with I^S. Here h and g are functions for inferring the rotation offset and generating synthetic images, and are approximated by CNNs.

It can be immediately seen that because CNNs are not equivariant to rotation, the performance of image generation in network g of Equation (1) is greatly dependent on the capability of network h, as any residual rotation offset between I_*^L and I^S will adversely affect the quality of image generation. Moreover, because the modality transfer requires a spatially proximal pair of I^L and I^S to have already been found, it has no capability to solve the retrieval (place recognition) problem.

We show in Section IV that by representing both I^L and I^S as 2D points, our method naturally handles the modality difference between the two domains, and learns a one-to-one mapping that is less ill-posed than image generation.

IV. Methodology

A. Learning Lidar Occupancy from Overhead Imagery

Given a satellite image I^S, we learn an occupancy image $\hat{\mathrm{O}}$ using a lidar image I^L pixel-aligned to I^S with ground truth pose for supervision. Specifically, $\hat{\mathrm{O}}$ should provide information on the likelihood of each pixel to cause a range return if a lidar situated near the image centroid takes a scan. The pixel values in $\hat{\mathrm{O}}$ are in the range $\begin{bmatrix}0, 1\end{bmatrix}$, where larger values indicate higher probability that a pixel is on "occupied" space, such as on buildings, and smaller values indicate "free-space", such as on streets and side-walks.

The training of $\hat{\mathrm{O}}$ is realised by taking into account the sensing nature of lidars. As also noted in [46], in a lidar image with no ground points, ray-tracing along each azimuth, pixels from the centroid to immediately in front of the first range return is likely to be on free-space. Pixels with range returns are likely to be on occupied space, for example on building façades. Moreover, for pixels with no return *and* situated behind the first return along their azimuth, we are *uncertain* if they are on occupied or free space. Using this knowledge, for each I^L we can construct a binary certainty mask M after thresholding I^L, as shown in Figure 4. Ray-tracing can be performed efficiently in parallel by transforming the lidar image to polar (range-azimuth) coordinate representation with a finite number of discretised azimuths.

Here M is a binary image where $\mathrm{M}(i,j) = 1$ if we are certain whether pixel $[i,j]$ is on occupied space or free-space, and $\mathrm{M}(i,j) = 0$ if we are uncertain about the occupancy of pixel $[i,j]$. Shown in Figure 4, in this process we have also created a binary lidar image $\mathrm{I}^{L\dagger}$, where $\mathrm{I}^{L\dagger}(i,j) = 1$ if $\mathrm{I}^L(i,j) > \gamma$ and $\mathrm{I}^{L\dagger}(i,j) = 0$ otherwise, and γ is a threshold we choose. In particular, we set γ to 0.2 in our experiments.

[2]A mapping $f : \mathcal{X} \rightarrow \mathcal{Y}$ is *equivariant* to a group of transformations Φ, if $\forall \phi \in \Phi$, $\exists \psi \in \Psi$ such that $\psi\big(f(\chi)\big) = f\big(\phi(\chi)\big), \forall \chi \in \mathcal{X}$.

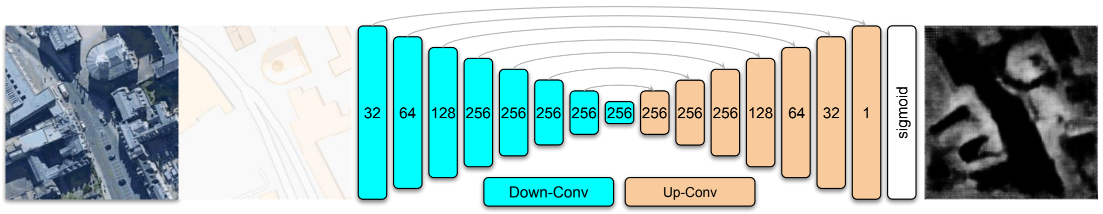

Fig. 3: To generate the occupancy image $\hat{\mathrm{O}}$, we use an U-Net architecture with 8 down-sample convolution blocks (each followed by Leaky ReLU and BatchNorm) and 8 up-sample convolution blocks (each followed by ReLU, BatchNorm, and dropout, except the last block), with skip connections between layers. We use 4×4 kernels with a stride of 2. The number on each block indicates the number of channels after the corresponding layer. The network takes in a 6-channel input (3 from I^S + 3 from I^R), and produces a 1-channel output, followed by a sigmoid.

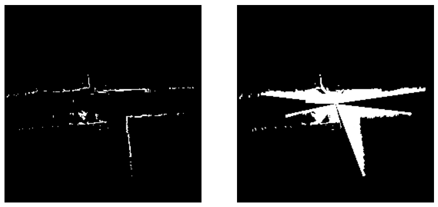

Fig. 4: Given a lidar image I^L we first threshold I^L to produce a binary lidar image $\mathrm{I}^{L\dagger}$ (left) where $\mathrm{I}^{L\dagger}(i,j)$ is 0 or 1 for any pixel $[i,j]$, with 1 indicating return and 0 indicating no return. Ray-tracing $\mathrm{I}^{L\dagger}$, pixels from the centroid until the first return in $\mathrm{I}^{L\dagger}$ along each azimuth are labelled as 1 in the corresponding certainty mask M (right). All pixels that are 1 in $\mathrm{I}^{L\dagger}$ are also labelled as 1 in M. All other pixels in M are labelled as 0.

Given I^S and I^L pixel-wise aligned using ground truth pose, we can learn an occupancy image as

$$f_o(\mathrm{I}^S) \rightarrow \hat{\mathrm{O}}, \tag{2}$$

where f_o is a function for generating $\hat{\mathrm{O}}$, parametrised by a deep network. The supervision signal comes from M and $\mathrm{I}^{L\dagger}$:

$$\begin{aligned}\mathcal{L}_{\text{occ}} = -\sum_{i,j} \mathrm{M}(i,j) \big[&\mathrm{I}^{L\dagger}(i,j) \cdot \log \hat{\mathrm{O}}(i,j) + \\ &\big(1 - \mathrm{I}^{L\dagger}(i,j)\big) \cdot \log \big(1 - \hat{\mathrm{O}}(i,j)\big) \big].\end{aligned} \tag{3}$$

The loss term in Equation (3) is a standard weighted binary cross entropy loss, where M are the weights and $\mathrm{I}^{L\dagger}$ are the labels. Notably, the loss only back-propagates through a small fraction of pixels where we are certain about the occupancy information, rather than through all pixels when using an L_1 loss for image generation as in RSL-Net [39]. By taking into account the nature of the sensor within the training procedure through the certainty mask M, our method naturally handles the occlusion and limited FOV of range sensors, and does not suffer from the ambiguity in learning a one-to-many mapping as described in Section III-B.

Finally, we found that overhead roadmap images, which are also publicly available from Google Maps, can provide additional semantic information that facilitates the learning of the occupancy image. Formally, we learn f_o as

$$f_o(\mathrm{I}^S, \mathrm{I}^R) \rightarrow \hat{\mathrm{O}}, \tag{4}$$

where I^R is a roadmap image queried at the same location as I^S. We parametrise f_o using a U-Net architecture [34], shown in Figure 3. We show in Figure 5 that at inference time when evaluated on unseen images, using I^R as an auxiliary input makes f_o more robust against regions in I^S with a strong pixel intensity gradient but do not correspond to a boundary between free and occupied space, such as shadows. We pretrain f_o for 100 epochs prior to downstream tasks.

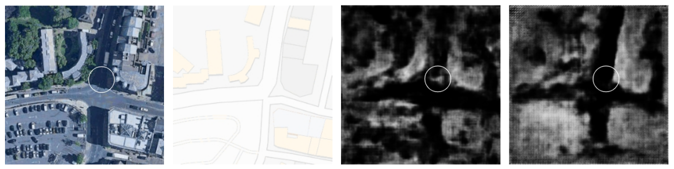

Fig. 5: Left: an unseen satellite image I^S at inference. Centre left: the corresponding I^R queried at the same location as I^S. The circled region corresponds to a strong gradient in I^S caused by shadows, and results in some pixels in this region inferred as occupied where they should be on free-space when we learn $f_o(\mathrm{I}^S) \rightarrow \hat{\mathrm{O}}$ (centre right). The occupancy image is more robust against shadows if we use I^R as an auxiliary input (right).

B. Points Extraction from Occupancy Image

Given the learned occupancy image $\hat{\mathrm{O}}$, we can emulate the resulting point-cloud as if a lidar takes a scan from near the centre of $\hat{\mathrm{O}}$, which we express as N 2D points where each point is a pixel coordinate. Similarly as when constructing M, we are confident that along each azimuth, the first pixel labelled as occupied space in $\hat{\mathrm{O}}$ is likely to cause a range return when scanned by a lidar. Nevertheless, it is *uncertain* whether the pixels behind the first occupied pixel will be "seen" by a lidar. As such, we ray-trace $\hat{\mathrm{O}}$, and keep only the first occupied pixel along each azimuth as a point return when forming the pseudo point-cloud. Our method for extracting a point-cloud from an occupancy image is detailed in Algorithm 1. In our experiments, we use images of size 256×256, and also set both N_A and N_R to 256, resulting in $\mathbf{P}^S \in \mathbb{R}^{256 \times 2}$.

C. Finding the Origin for Ray-Tracing

By default, Algorithm 1 ray-traces from the centroid of $\hat{\mathrm{O}}$. A degenerate solution occurs if the centroid is on occupied space, in which case the value of the first pixel in every azimuth is larger than γ, and Algorithm 1 returns a circular point-cloud.

Given an $h \times h$ patch around the centroid of $\hat{\mathrm{O}}$, we can compute the distance of a pixel within the patch to its nearest occupied pixel. Then, a pixel that is furthest away from its nearest occupied pixel is likely to be on free-space and not enclosed by occupied pixels. Denote such pixel location as $[i^*, j^*]$, proper ray-tracing can be ensured if we use $[i^*, j^*]$ as the origin, rather than the image centroid of $\hat{\mathrm{O}}$. Figure 6 is a simple drawing that depicts a 10×10 patch where the centroid is on occupied space, and the distance of a particular pixel (cyan) to its nearest occupied pixel is traced. Such a process of finding the origin for ray-tracing, as well as the

Algorithm 1: Forming a pseudo point-cloud

```
Input:
$\hat{\mathrm{O}}$    // learned occupancy image of size $H \times H$
Parameters:
$N_A$    // number of discretised azimuths
$N_R$    // number of discretised ranges
$\gamma$    // pixel intensity threshold for occupancy
Output:
$\mathbf{P}^S \in \mathbb{R}^{N_A \times 2}$    // 2D point-cloud
$\mathbf{s} \in \mathbb{R}^{N_A}$    // scores for registration
Initialise:
$\mathbf{P}^S \leftarrow \mathbf{0}_{N_A \times 2}$    // assign $\begin{bmatrix}0 & 0\end{bmatrix}$ if no range return
$\mathbf{s} \leftarrow \mathbf{0}_{N_A}$    // assign a score of 0 if no range return
$\hat{\mathrm{O}}^p \leftarrow \hat{\mathrm{O}}$    // transform $\hat{\mathrm{O}}$ to range-azimuth
// representation resulting in $N_R \times N_A$ polar image $\hat{\mathrm{O}}^p$
for $i = 1, 2, \cdots, N_A$ do
    for $j = 1, 2, \cdots, N_R$ do
        if $\hat{\mathrm{O}}^p(i, j) \geq \gamma$ then
            $\mathbf{P}^S(i, :) \leftarrow$
            $\begin{bmatrix} j\frac{H}{2N_R}\cos(i\frac{2\pi}{N_A}) & j\frac{H}{2N_R}\sin(i\frac{2\pi}{N_A}) \end{bmatrix}$
            $\mathbf{s}(i) \leftarrow 1$
            break    // break and iterate to the next $i$
```

ray-tracing process itself in Algorithm 1, can be conducted efficiently in parallel with tensor operations.

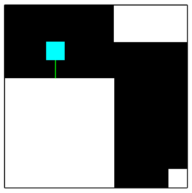

Fig. 6: A 10×10 image patch where the centroid is on occupied space, which are labelled as white pixels. For a particular pixel in this patch (cyan), we can compute its distance (green) to the nearest occupied pixel. The pixel within the patch that is the furthest away from its nearest occupied pixel is chosen as the origin for ray-tracing in Algorithm 1.

D. Learning Pose Estimation

Given a lidar image I^L spatially proximal to I^S, we can extract a point-cloud $\mathbf{P}^L \in \mathbb{R}^{N_A \times 2}$ using Algorithm 1 with I^L rather than $\hat{\mathrm{O}}$ as the input. In this process, only the first range return along each azimuth is kept in $\mathbf{P}^L$, making $\mathbf{P}^L$ more compatible to $\mathbf{P}^S$, since $\mathbf{P}^S$ has only one point per azimuth.

Given $\mathbf{P}^L$ and $\mathbf{P}^S$ spatially proximal but with an unknown $SE(2)$ offset, we use DCP [44] to solve for their pose difference. DCP was shown to outperform ICP on non-identical point sets [44], and in addition learns descriptors that can be used later for place recognition. DCP utilises DGCNN [45] and a Transformer [42] module to compute a d-dimensional descriptor per point, performs a soft matching, and uses SVD (as detailed in [37]) for pose estimation. When computing the covariance matrix in SVD, we weigh each correspondence using the score vector $\mathbf{s}$ from Algorithm 1, such that azimuths in $\hat{\mathrm{O}}$ with no return do not contribute to pose estimation.

DCP outputs an estimated rotation matrix $\hat{\mathbf{R}} \in \mathbb{R}^{2\times 2}$ and a translation vector $\hat{\mathbf{t}} \in \mathbb{R}^2$. We can establish a loss term using ground truth $\mathbf{R}$ and $\mathbf{t}$:

$$\mathcal{L}_{\text{pose}} = \left\|\hat{\mathbf{t}} - \mathbf{t}\right\|_1 + \lambda \left\|\hat{\mathbf{R}}\mathbf{R}^T - \mathbf{I}\right\|_1, \tag{5}$$

where $\mathbf{I}$ is the identity matrix and λ is a relative weighting between the rotational and the translational components.

We use an L_1 loss rather than L_2 loss as in [44], since L_1 loss is more robust against the non-identical point-clouds $\mathbf{P}^L$ and $\mathbf{P}^S$ in our case. We set $\lambda = 10$ in our experiments. The points extraction step in Algorithm 1 is fully differentiable, allowing the loss in Equation (5) to not only optimise the DCP network, but also fine-tune the pretrained network for f_o, without needing to also apply the loss in Equation (3). The data flow for pose estimation is shown on the left of Figure 7.

E. Learning Place Recognition

After the networks are trained for pose estimation, the DGCNN+Transformer module in DCP is optimised for producing local descriptors useful for registration. Given a point-cloud $\mathbf{P} \in \mathbb{R}^{N\times 2}$ and its local descriptors $\mathbf{D} \in \mathbb{R}^{N\times d}$, we use a PointNetVLAD [41] layer to learn a k-dimensional global descriptor $\mathbf{d}$, shown on the right of Figure 7.

To optimize the PointNetVLAD layer, we fix the network parameters for f_o and DCP, and apply a triplet loss using descriptors from positive (spatially proximal) and negative (spatially distant) pairs of $\mathbf{P}^S$ and $\mathbf{P}^L$. Formally, this is

$$\mathcal{L}_{\text{pr}} = \left[\left\|\mathbf{d}^{S+} - \mathbf{d}^L\right\|_2 - \left\|\mathbf{d}^{S-} - \mathbf{d}^L\right\|_2 + m\right]_+, \tag{6}$$

where m is the triplet margin which we set as 1, $[a]_+$ denotes $\max(a, 0)$, and $\mathbf{d}^L$ is the global descriptor for point-cloud $\mathbf{P}^L$ from a lidar image I^L. $\mathbf{d}^{S+}$ is the global descriptor for pseudo point-cloud $\mathbf{P}^{S+}$ from a pair of satellite and roadmap images I^{S+} and I^{R+} queried at a spatially proximal location as I^L. $\mathbf{d}^{S-}$ is the global descriptor from I^{S-} and I^{R-} spatially distant and have no geometric overlap with I^L.

V. Experimental Results

We validate our method using the Oxford Radar RobotCar Dataset [4] and the KITTI Raw Dataset [18]. The lidar data is collected using Velodyne HDL-32E for RobotCar and Velodyne HDL-64E for KITTI. Both datasets have longitude/latitude data for each time-stamp to query for overhead images. The resolution for $\mathrm{I}^S, \mathrm{I}^R$, and I^L are 0.4332 m/pixel in the RobotCar dataset and 0.4592 m/pixel in KITTI. We use images of size 256×256 in all of our experiments.

Our method is the first to learn the place recognition of a lidar using only overhead imagery, and we benchmark against a baseline method that trains a VGG-16 network [36] with BatchNorm followed by a NetVLAD [1] layer directly on the lidar and overhead images, using triplet loss. For pose estimation, we compare against recent work [39] on learning the metric localisation between range sensors and satellite imagery. For a fair comparison, our choice of training, validation, and test trajectories in each dataset is made as close to the lidar experiments in [40] as possible. Specifically,

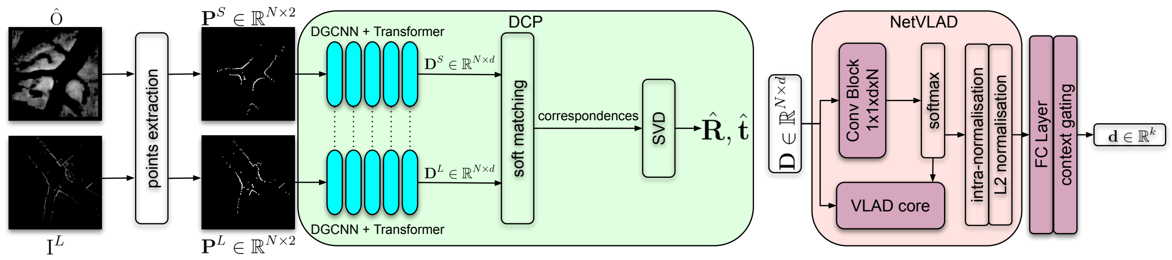

Fig. 7: Data flow at forward pass. Left: for pose estimation, given a spatially proximal pair of $\hat{\mathrm{O}}$ and I^L, we extract point-clouds $\mathbf{P}^S$ and $\mathbf{P}^L$ respectively, using Algorithm 1. The DGCNN+Transformer module in DCP learns local descriptors for each point-cloud, which are then used for soft matching to establish correspondences. The final SVD module in DCP returns the estimated pose offset between $\mathbf{P}^S$ and $\mathbf{P}^L$. Right: for place recognition, given local descriptors $\mathbf{D} \in \mathbb{R}^{N \times d}$ for a point-cloud $\mathbf{P} \in \mathbb{R}^{N \times 2}$, we use PointNetVLAD to produce a global descriptor $\mathbf{d} \in \mathbb{R}^k$, which can be used for retrieval.

for RobotCar, the training trajectories in sequences no.2, no.5, and no.6 are used for training while we only validate and test on no.2. For KITTI, the training data consists of sequences `20110929_drive0071`, `20110930_drive0028`, `20111003_drive0027`, and `20111003_drive0034`, where `20111003_drive0034` is split into training and test trajectories. `20110926_drive0117` forms the validation set. The test trajectory lengths are approximately 1.26 km for RobotCar and 1.44 km for KITTI. The data splits can be visualised in Figure 8.

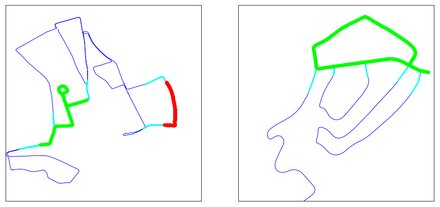

Fig. 8: Trajectories are split into training (blue), validation (red), and test (green) for all sequences in RobotCar (left) and `20111003_drive0034` in KITTI (right). Regions near the intersections between splits are discarded (cyan) to avoid any overlap between training, validation, and test trajectories.

The training takes three stages. We first pretrain f_o for 100 epochs using the loss in Equation (3), where we apply random rotations for data augmentation. We then train f_o and the DCP network end-to-end on pose estimation, using the pose loss in Equation (5). Finally, we fix the network parameters of f_o and DCP, and optimise a PointNetVLAD layer for place recognition using the triplet loss in Equation (6). We use a fixed learning rate of 2×10^{-4} in stages 1 and 3, and 1×10^{-4} in stage 2. We train stages 2 and 3 for 140 epochs and choose the check-point with the best validation set performance.

A. Metric Localisation

To evaluate metric localisation, we assume place recognition is solved such that we have spatially proximal pairs of I^L and I^S (and I^R) where their initial pose offset is no larger than a certain amount. Our method then registers point-clouds $\mathbf{P}^S$ and $\mathbf{P}^L$ to solve for a relative $SE(2)$ pose offset.

We compare against RSL-Net [39] where the initial pose offset is a uniform distribution in each of x, y, and θ within a certain range. Specifically, we consider initial offsets that are large in both rotation and translation, large in rotation but small in translation, and large in translation but small in rotation. The mean metric localisation errors for various initial pose offsets are summarised in Tables I and II.

In the original work [39, 40], RSL-Net was only evaluated on experiments with small rotation offset in the range of $[-22.5°, 22.5°]$ where it shows low errors; yet, we show its accuracy degrades rapidly for larger initial offsets, and it is outperformed by our method. In terms of inference time, the forward pass for all modules takes approximately 0.075 s altogether on a 1080Ti GPU, running approximately 25% faster than RSL-Net which runs at 10 Hz.

1) Introspection: For overhead imagery situated along a narrow, straight road, the pseudo point-cloud $\mathbf{P}^S$ can be roughly symmetrical along its principal direction, shown in Figure 9. In such cases, our method sometimes outputs an estimated $\hat{\mathbf{R}}$ that is approximately 180° off from the true offset as such a solution is equally "correct" for symmetrical point-clouds. The point-cloud registration problem is ill-posed if symmetry exists along at least one direction, in particular if the initial rotation offset can be anywhere from $-\pi$ to π.

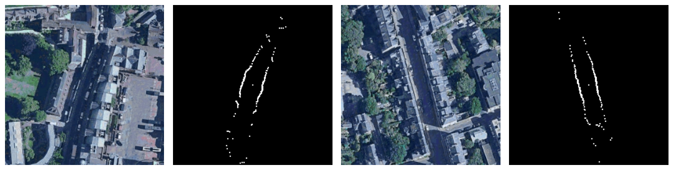

Fig. 9: Satellite images I^S (1 and 3) and the corresponding pseudo point-clouds $\mathbf{P}^S$ (2 and 4) on narrow, straight roads show a strong symmetry.

We can introspect whether $\mathbf{P}^S$ is a symmetrical point-cloud by taking the Chamfer distance after a rotation by π :

$$\mathrm{d}_{\text{chamfer}} = \sum_i \mathrm{d}_{\min}(\mathbf{p}_i^S, \mathbf{R}_\pi \mathbf{P}^S), \tag{7}$$

where each $\mathbf{p}_i^S \in \mathbb{R}^2$ is a point in $\mathbf{P}^S$, $\mathrm{d}_{\min}(\mathbf{p}, \mathbf{P})$ is the minimum distance from a point $\mathbf{p}$ to any point in point-cloud $\mathbf{P}$, and $\mathbf{R}_\pi \in \mathbb{R}^{2\times 2}$ is a rotation by π. While such an introspection methodology only considers symmetries of π, man-made structures in practice tend to result in a π symmetry.

Given γ_{cd}, a threshold on the Chamfer distance, we can treat a solution as "confident" if $\mathbf{P}^S$ results in a Chamfer distance from Equation (7) that is larger than γ_{cd}, in which case $\mathbf{P}^S$ is regarded as non-symmetrical. For initial offsets for x, y, and θ in the range ± 10 pixels, ± 10 pixels, and $\pm 180°$ respectively, Figure 10 shows the mean θ error for solutions considered as confident, and the percentage of solutions considered as confident, evaluated on RobotCar, for various values of γ_{cd}.

Initial offset for x, y, and θ (pixel, pixel, $^\circ$)	$\pm25, \pm25, \pm180$			$\pm10, \pm10, \pm180$			$\pm25, \pm25, \pm90$			$\pm10, \pm10, \pm90$			$\pm25, \pm25, \pm45$			$\pm25, \pm25, \pm22.5$		
	x	y	θ	x	y	θ	x	y	θ	x	y	θ	x	y	θ	x	y	θ
RSL-Net [39]	**7.45**	**9.51**	68.53	6.08	8.05	58.69	12.45	13.03	34.79	5.90	6.81	10.75	**5.85**	**7.04**	**2.36**	**5.33**	**5.89**	**2.08**
Ours	9.77	10.28	**49.18**	**4.84**	**4.77**	**25.36**	**7.43**	**10.18**	**11.41**	**4.39**	**4.13**	**7.16**	6.34	7.85	5.53	5.89	7.54	3.50

TABLE I: Lidar metric localisation errors for various ranges of initial offset, evaluated on RobotCar. The units are pixels for x and y and degrees for θ.

Initial offset for x, y, and θ (pixel, pixel, $^\circ$)	$\pm25, \pm25, \pm180$			$\pm10, \pm10, \pm180$			$\pm25, \pm25, \pm90$			$\pm10, \pm10, \pm90$			$\pm25, \pm25, \pm45$			$\pm25, \pm25, \pm22.5$		
	x	y	θ	x	y	θ	x	y	θ	x	y	θ	x	y	θ	x	y	θ
RSL-Net [39]	7.29	11.93	70.93	5.33	7.45	67.28	**5.62**	9.49	11.19	4.56	7.61	8.36	5.54	9.93	**2.56**	5.34	**6.08**	**1.59**
Ours	**6.91**	**11.22**	**62.79**	**3.71**	**4.50**	**34.51**	5.89	**8.89**	**9.10**	**3.44**	**4.41**	**6.73**	**5.14**	**6.97**	4.52	**5.08**	7.15	3.70

TABLE II: Lidar metric localisation errors for various ranges of initial offset, evaluated on KITTI. The units are pixels for x and y and degrees for θ.

Initial offset for x, y, and θ (pixel, pixel, $^\circ$)	$\pm25, \pm25, \pm180$			$\pm10, \pm10, \pm180$			$\pm25, \pm25, \pm90$			$\pm10, \pm10, \pm90$			$\pm25, \pm25, \pm45$			$\pm25, \pm25, \pm22.5$		
	x	y	θ	x	y	θ	x	y	θ	x	y	θ	x	y	θ	x	y	θ
RSL-Net [39]	**7.11**	**8.49**	62.78	**3.27**	**4.69**	59.04	7.86	**8.27**	**8.51**	**3.43**	**4.30**	13.73	7.17	8.48	**3.80**	**6.47**	**7.45**	**2.90**
Ours	8.47	11.38	**49.62**	4.86	5.38	**35.90**	**7.41**	9.55	17.02	4.10	4.64	**13.00**	**6.25**	**8.07**	7.44	6.55	7.78	5.59

TABLE III: Radar metric localisation errors for various ranges of initial offset, evaluated on RobotCar. The units are pixels for x and y and degrees for θ.

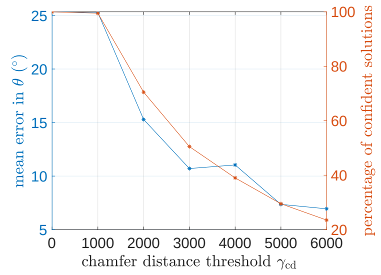

Fig. 10: Percentage of solutions considered "confident" and the mean θ error of confident solutions, for various values of γ_{cd}, evaluated on RobotCar.

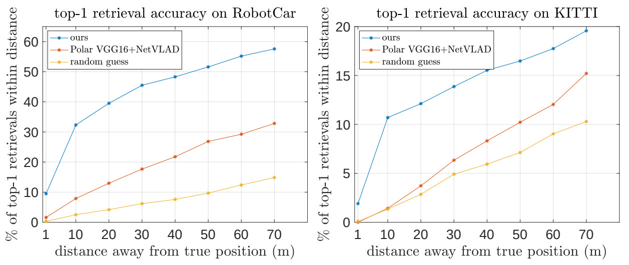

Fig. 11: Percentage of top-1 retrievals falling within a certain distance away from the true position.

B. Place Recognition

In this experiment, we first query the overhead images using the GPS longitude/latitude at each time-stamp during the test route traversal. In a more realistic scenario, while the routes to be evaluated can be known beforehand to gather overhead images ahead of time and use as a prior map, the exact driving path may be different in the live traversal, for example due to driving on different lanes along the same road. To simulate this, we add uniform errors in the range $[-5 \text{ m}, 5 \text{ m}]$ to each of the x and y direction when querying for overhead images.

Other than the modality difference, a major challenge when compared against standard lidar-to-lidar place recognition is the arbitrary rotation offset between lidar scans and overhead imagery. In standard lidar-based place recognition, because the vehicle will likely face the same (or exactly opposite) direction when driving along the same route, spatially proximal lidar scans from two traversals will have little rotation offset (or approximately π). However, overhead images are expressed in a fixed orientation (North-up), while lidar scans in their local reference frame do not have a privileged orientation. As such, without another sensor (e.g., a magnetometer) to measure the global heading, spatially proximal lidar and overhead image pairs can in fact be offset by an arbitrary rotation, increasing the comparison complexity. To foster robustness to rotation, we apply random rotations in the range $[-\pi, \pi]$ between $\mathbf{P}^S$ and $\mathbf{P}^L$ when learning local descriptors, and between $\mathbf{P}^{S+}$ and $\mathbf{P}^L$ when training the PointNetVLAD layer.

1) Top-1 Accuracy: Given a database of all pairs of satellite and roadmap images $\{\mathrm{I}_i^S, \mathrm{I}_i^R\}, i = 1, \cdots, N$ queried along the test trajectory, we form a pseudo point-cloud and output a global descriptor for each sample, resulting in a database of map global descriptors $\{\mathbf{d}_i^S\}$. For some lidar image I_j^L taken along the test trajectory and its associated global descriptor $\mathbf{d}_j^L$, the top-1 retrieval is the closest $\mathbf{d}_i^S \in \{\mathbf{d}_i^S\}$ in terms of Euclidean distance. Figure 11 shows the percentage of top-1 match within certain distances away from the true position. We compare against a baseline method that feeds $\{\mathrm{I}^S, \mathrm{I}^R\}$ and I^L to a VGG16+NetVLAD network and learns a triplet loss, where the best baseline performance comes from representing all input as polar images prior to passing through the network, and using two different networks for the two sources.

On the RobotCar Dataset, which features a city environment, our method achieves a high accuracy, having over half of all top-1 retrievals falling within 50 m of the true position. The test set for KITTI, featuring a residential area, is much more challenging as residential areas have more structurally repetitive and fewer geometrically distinctive places. Regardless, our method consistently outperforms the baseline method.

2) Failure Cases: Overhead images queried at different locations may result in structurally similar point-clouds $\mathbf{P}^S$, leading to false top-1 retrievals, as depicted in Figure 13.

3) Precision and Recall: To evaluate precision and recall, we sample a threshold γ_{desc} uniformly from the minimum Euclidean distance between any $\mathbf{d}_i^S$ to any $\mathbf{d}_j^L$, to the maximum. A pair $\mathbf{d}_i^S$ and $\mathbf{d}_j^L$ is considered a positive retrieval if their Euclidean distance is within γ_{desc}, and otherwise negative. Two locations are considered a true match if their distance is within 25 m, and false match if it is greater than 50 m. We consistently outperform the baseline as shown in Figure 14.

C. Extending to Radar Localisation using Overhead Imagery

Our method relies on the premise that in a lidar image, pixels from the centroid to the first return along each azimuth is considered free-space. This is not a valid assumption for scanning radar images that are prone to speckle interference.

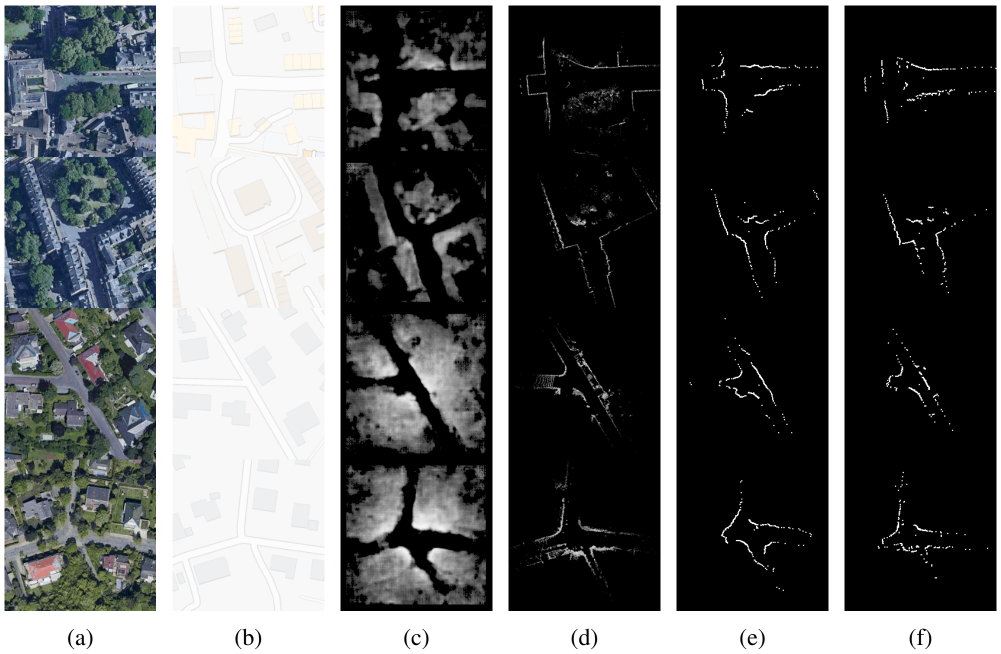

Fig. 12: Qualitative results from the test set of RobotCar (rows 1 and 2) and KITTI (rows 3 and 4). From left to right: satellite image I^S (a), roadmap image I^R (b), occupancy image $\hat{\mathrm{O}}$ (c), lidar image I^L (d), point-clouds $\mathbf{P}^S$ (e) and $\mathbf{P}^L$ (f). All images are pose-aligned for better visualisation.

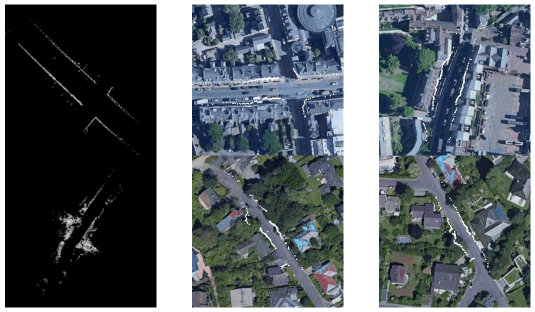

Fig. 13: Failure cases: a lidar image I^L_j along the test route (left), the satellite image and the associated pseudo point-cloud corresponding to the true position (centre), and the falsely retrieved satellite image and its associated pseudo point-cloud (right), evaluated on RobotCar (top) and KITTI (bottom).

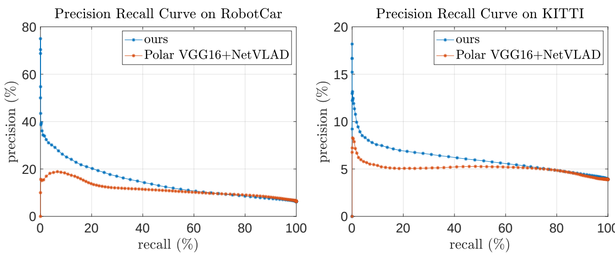

Fig. 14: Precision and recall curve for various descriptor distance thresholds.

However, we can reliably transform radar images to the appearance of lidar images using unpaired image-to-image transfer [55], specifically the methodology in [47].

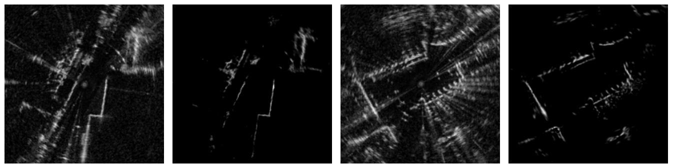

Fig. 15: Radar images (1 and 3) transformed into the appearance of lidar images (2 and 4) using the Cyclegan implementation in [47].

Figure 15 shows the synthetic lidar images created from radar images. Once radar images are transformed into the appearance of lidar images, they are used as input in our method as usual for training and inference. We evaluate on the RobotCar Dataset, which also has data from a Navtech radar, and use the same sequences and splits as in the RobotCar lidar experiments. The metric localisation and place recognition results are shown in Table III and Figure 16.

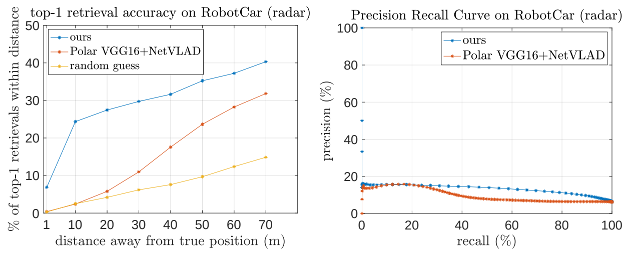

Fig. 16: Place recognition results using radar.

D. *Additional Qualitative Results*

Additional qualitative results are shown in Figure 12.

VI. Conclusion and Future Work

In this paper we introduce a novel method that solves both place recognition and metric localisation of lidar using only publicly available overhead imagery. Specifically, the modality difference is handled in a natural way by representing both data sources as point sets. While our method is the first to learn the place recognition of a ground lidar using overhead imagery, it also outperforms a prior method on metric localisation when the initial pose offset is large. While the method as detailed in this paper is standalone and requires only overhead imagery, a future work would be a large-scale lidar localisation system with prior lidar maps, where solutions using overhead imagery are used as auxiliary signals to further improve accuracy.

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016.

[2] Ian Baldwin and Paul Newman. Road Vehicle Localization with 2D Push-broom LIDAR and 3D Priors. In *2012 IEEE International Conference on Robotics and Automation*, pages 2611–2617. IEEE, 2012.

[3] Dan Barnes and Ingmar Posner. Under the Radar: Learning to Predict Robust Keypoints for Odometry Estimation and Metric Localisation in Radar. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9484–9490. IEEE, 2020.

[4] Dan Barnes, Matthew Gadd, Paul Murcutt, Paul Newman, and Ingmar Posner. The Oxford Radar Robotcar dataset: A Radar Extension to the Oxford Robotcar Dataset. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6433–6438. IEEE, 2020.

[5] Ioan Andrei Barsan, Shenlong Wang, Andrei Pokrovsky, and Raquel Urtasun. Learning to Localize Using a LiDAR Intensity Map. In *CoRL*, pages 605–616, 2018.

[6] Paul J Besl and Neil D McKay. Method for Registration of 3-D Shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

[7] Michael Bosse and Robert Zlot. Place Recognition Using Keypoint Voting in Large 3D Lidar Datasets. In *2013 IEEE International Conference on Robotics and Automation*, pages 2677–2684. IEEE, 2013.

[8] Marcus A Brubaker, Andreas Geiger, and Raquel Urtasun. Lost! Leveraging the Crowd for Probabilistic Visual Self-localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3057–3064, 2013.

[9] Patrick JF Carle and Timothy D Barfoot. Global Rover Localization by Matching Lidar and Orbital 3D Maps. In *2010 IEEE International Conference on Robotics and Automation*, pages 881–886. IEEE, 2010.

[10] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep Global Registration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2514–2523, 2020.

[11] Hang Chu, Hongyuan Mei, Mohit Bansal, and Matthew R Walter. Accurate Vision-Based Vehicle Localization Using Satellite Imagery. *arXiv preprint arXiv:1510.09171*, 2015.

[12] Konrad P Cop, Paulo VK Borges, and Renaud Dubé. Delight: An Efficient Descriptor for Global Localisation Using lidar Intensities. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3653–3660. IEEE, 2018.

[13] Lucas de Paula Veronese, Edilson de Aguiar, Rafael Correia Nascimento, Jose Guivant, Fernando A Auat Cheein, Alberto Ferreira De Souza, and Thiago Oliveira-Santos. Re-emission and Satellite Aerial Maps Applied to Vehicle Localization on Urban Environments. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4285–4290. IEEE, 2015.

[14] Can Ulas Dogruer, A Bugra Koku, and Melik Dolen. Outdoor Mapping and Localization Using Satellite Images. *Robotica*, 28(7):1001–1012, 2010.

[15] Juan Du, Rui Wang, and Daniel Cremers. DH3D: Deep Hierarchical 3D Descriptors for Robust Large-Scale 6DoF Relocalization. In *European Conference on Computer Vision*, pages 744–762. Springer, 2020.

[16] Renaud Dubé, Daniel Dugas, Elena Stumm, Juan Nieto, Roland Siegwart, and Cesar Cadena. Segmatch: Segment Based Place Recognition in 3d Point Clouds. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5266–5272. IEEE, 2017.

[17] Georgios Floros, Benito Van Der Zander, and Bastian Leibe. OpenStreetSLAM: Global Vehicle Localization Using OpenStreetMaps. In *2013 IEEE International Conference on Robotics and Automation*, pages 1054–1059. IEEE, 2013.

[18] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision Meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[19] Jiadong Guo, Paulo VK Borges, Chanoh Park, and Abel Gawel. Local Descriptor for Robust Place Recognition Using Lidar Intensity. *IEEE Robotics and Automation Letters*, 4(2):1470–1477, 2019.

[20] Marwan Hussein, Matthew Renner, Masaaki Watanabe, and Karl Iagnemma. Matching of Ground-Based LiDAR and Aerial Image Data for Mobile Robot Localization in Densely Forested Environments. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1432–1437. IEEE, 2013.

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[22] Dong-Ki Kim and Matthew R Walter. Satellite Image-Based Localization via Learned Embeddings. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2073–2080. IEEE, 2017.

[23] Giseop Kim and Ayoung Kim. Scan Context: Egocentric Spatial Descriptor for Place Recognition within 3d Point Cloud Map. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4802–4809. IEEE, 2018.

[24] Giseop Kim, Byungjae Park, and Ayoung Kim. 1-Day Learning, 1-Year Localization: Long-Term Lidar Localization Using Scan Context Image. *IEEE Robotics and Automation Letters*, 4(2):1948–1955, 2019.

[25] Rainer Kümmerle, Bastian Steder, Christian Dornhege,

Alexander Kleiner, Giorgio Grisetti, and Wolfram Burgard. Large Scale Graph-based SLAM using Aerial Images as Prior Information. *Autonomous Robots*, 30 (1):25–39, 2011.

[26] Karel Lenc and Andrea Vedaldi. Understanding Image Representations by Measuring Their Equivariance and Equivalence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 991–999, 2015.

[27] Keith Yu Kit Leung, Christopher M Clark, and Jan P Huissoon. Localization in Urban Environments by Matching Ground Level Video Images with an Aerial Image. In *2008 IEEE International Conference on Robotics and Automation*, pages 551–556. IEEE, 2008.

[28] Zhe Liu, Shunbo Zhou, Chuanzhe Suo, Peng Yin, Wen Chen, Hesheng Wang, Haoang Li, and Yun-Hui Liu. LPD-Net: 3D Point Cloud Learning for Large-Scale Place Recognition and Environment Analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2831–2840, 2019.

[29] Masafumi Noda, Tomokazu Takahashi, Daisuke Deguchi, Ichiro Ide, Hiroshi Murase, Yoshiko Kojima, and Takashi Naito. Vehicle Ego-Localization by Matching in-Vehicle Camera Images to an Aerial Image. In *Asian Conference on Computer Vision*, pages 163–173. Springer, 2010.

[30] Pilailuck Panphattarasap and Andrew Calway. Automated Map Reading: Image Based Localisation in 2-d Maps Using Binary Semantic Descriptors. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6341–6348. IEEE, 2018.

[31] Oliver Pink. Visual Map Matching and Localization Using a Global Feature Map. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–7. IEEE, 2008.

[32] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[33] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv preprint arXiv:1706.02413*, 2017.

[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[35] Turgay Senlet and Ahmed Elgammal. A Framework for Global Vehicle Localization Using Stereo Images and Satellite and Road Maps. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2034–2041. IEEE, 2011.

[36] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proceedings of the 3rd International Conference on Learning Representations(ICLR)*, 2015.

[37] Olga Sorkine-Hornung and Michael Rabinovich. Least-Squares Rigid Motion Using SVD. *Computing*, 1(1):1–5, 2017.

[38] Li Sun, Daniel Adolfsson, Martin Magnusson, Henrik Andreasson, Ingmar Posner, and Tom Duckett. Localising Faster: Efficient and Precise Lidar-Based Robot Localisation in Large-Scale Environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4386–4392. IEEE, 2020.

[39] T. Y. Tang, D. De Martini, D. Barnes, and P. Newman. RSL-Net: Localising in Satellite Images From a Radar on the Ground. *IEEE Robotics and Automation Letters*, 5(2):1087–1094, April 2020. ISSN 2377-3774. doi: 10.1109/LRA.2020.2965907.

[40] Tim Y Tang, Daniele De Martini, Shangzhe Wu, and Paul Newman. Self-Supervised Localisation between Range Sensors and Overhead Imagery. In *Robotics: Science and Systems (RSS) XVI*, 2020.

[41] Mikaela Angelina Uy and Gim Hee Lee. PointNetVLAD: Deep Point Cloud Based Retrieval for Large-Scale Place Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4470–4479, 2018.

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, 2017.

[43] Yue Wang and Justin Solomon. PRNet: Self-Supervised Learning for Partial-to-Partial Registration. *NeurIPS*, 2019.

[44] Yue Wang and Justin M Solomon. Deep Closest Point: Learning Representations for Point Cloud Registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3523–3532, 2019.

[45] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic Graph CNN for Learning on Point Clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[46] Rob Weston, Sarah Cen, Paul Newman, and Ingmar Posner. Probably Unknown: Deep Inverse Sensor Modelling Radar. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5446–5452. IEEE, 2019.

[47] Rob Weston, Oiwi Parker Jones, and Ingmar Posner. There and Back Again: Learning to Simulate Radar Data for Real-World Applications. *arXiv preprint arXiv:2011.14389*, 2020.

[48] Ryan W Wolcott and Ryan M Eustice. Fast LIDAR Localization Using Multiresolution Gaussian Mixture Maps. In *2015 IEEE international Conference on Robotics and Automation (ICRA)*, pages 2814–2821. IEEE, 2015.

[49] Fan Yan, Olga Vysotska, and Cyrill Stachniss. Global Localization on OpenStreetMap Using 4-bit Semantic Descriptors. In *2019 European Conference on Mobile*

Robots (ECMR), pages 1–7. IEEE, 2019.

[50] Zi Jian Yew and Gim Hee Lee. 3DFeat-Net: Weakly Supervised Local 3D Features for Point Cloud Registration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 607–623, 2018.

[51] Huan Yin, Yue Wang, Xiaqing Ding, Li Tang, Shoudong Huang, and Rong Xiong. 3D Lidar-Based Global Localization Using Siamese Neural Network. *IEEE Transactions on Intelligent Transportation Systems*, 21(4):1380–1392, 2019.

[52] Wentao Yuan, Benjamin Eckart, Kihwan Kim, Varun Jampani, Dieter Fox, and Jan Kautz. DeepGMR: Learning Latent Gaussian Mixture Models for Registration. In *European Conference on Computer Vision*, pages 733–750. Springer, 2020.

[53] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1802–1811, 2017.

[54] Wenxiao Zhang and Chunxia Xiao. PCAN: 3D Attention Map Learning Using Contextual Information for Point Cloud Based Retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12436–12445, 2019.

[55] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.