Localisation of Ground Range Sensors Using Overhead Imagery



Tim Yuqing Tang

St.Cross College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Hilary 2023

Acknowledgements

Personal

I am extremely grateful to my parents' decision to move aboard and their continued, selfless care and support over the years.

The research outcomes of this thesis would not have been possible without the aid of many. My academic advisor, Professor Paul Newman is extraordinarily helpful with my studies, and is always there to provide endless interesting ideas and suggestions with his incomparable knowledge in robotics. When I make mistakes, which has happened frequently during the course of my DPhil, Paul is always patient, encouraging, and eventually carries me back on track. Paul is the most thoughtful advisor one can ask for and one who truly cares about the well-being of his students. Completing a DPhil through a global pandemic was tough in many ways, but Paul has made my time at Oxford well enjoyable and memorable.

Help and support from my post-doc mentor, Daniele De Martini, is indispensable to my thesis work. Daniele gave countless valuable feedbacks during our regular meetings, and is tremendously supportive in providing guidance on lower-level algorithms and methodologies and implementation details. The writings in our publications were substantially improved after Daniele's careful proof-reads. I would like to express my gratitude to my collaborators, Dan Barnes and Shangzhe Wu, who have taught me a lot during our meetings and conversations. I am sincerely grateful to my colleagues at Oxford, Matt Gadd, Ştefan Săftescu, Rob Weston, Horia Porav, Yizhe Wu, Zirui Wang, Roberto Aldera, Tom Bruls, and Valentina Musat for idea discussions. Also, thanks to Professors Ingmar Posner and Victor Prisacariu for charing my Transfer and Confirmation of Studies, and your valuable feedback and guidance on my work.

I am indebted to my previous advisors at the University of Toronto for carrying me along my academic path. The opportunity to study a DPhil at Oxford would not have been possible without the support and teachings from my Master's supervisor, Professor Tim Barfoot, who was not only a superb research advisor, but also a kind and responsible life mentor. I would like to thank Professors Jonathan Kelly and Angela Schoellig, for introducing me to the world of robotics research.

I was fortunate to complete an internship opportunity at Meta Reality Labs Research, where I got the chance to work with a team of world-class scientists and

learn about how robotics and computer vision are applied to real-world problems. I would like to thank my manager Hyo Jin Kim and mentors Tianwei Shen, Meng Tang, Vincent Lee, Shichao Yang, Tomasz Malisiewicz, Chris Sweeney, and Richard Newcombe for making this an unforgettable experience.

Finally, thank you Angela, for bringing joy to my life.

Institutional

This DPhil is financially supported by an Oxford Robotics Institute Scholarship and a PGS-D award from NSERC Canada.

Abstract

This thesis is about outdoor localisation using range sensors as an active sensor and cheap, publicly available satellite or 'overhead' imagery as a prior map. Range sensors such as lidars and spinning FMCW radars are ideal for large-scale, outdoor autonomous navigation due to their long sensing range, invariance to lighting conditions, and robustness against weather changes. Nevertheless, existing methods for range sensor localisation typically rely on prior maps collected from a previous mapping phase. On the other hand, off-the-shelf overhead imagery, such as public satellite images, is readily available almost anywhere in the world and can be acquired easily from the internet with little cost or effort.

Public overhead imagery can capture geometric cues of the scene also observable by ground lidars and radars, therefore having the capability to act as a map for range sensor localisation. In particular, in corner case scenarios where the prior sensory map is unusable or unavailable, for example if the robot travels to a place it has not visited before, public overhead imagery can act as an alternative map source for range sensor localisation as a fall-back choice. Under normal operation conditions, the localisation result by comparing range sensor data against overhead imagery can act as an additional information source for redundancy.

In this thesis, we present various methods to solve the localisation of a ground range sensor using overhead imagery by learning from data, enabling them to adapt to different environments. This surpasses the methods in literature which employ hand-crafted features designed for only specific types of scenery. Specifically, we address both topological localisation, also known as place recognition, and metric localisation in overhead imagery maps. Furthermore, we investigate self-supervised strategies that allow the tasks to be learned without accurate ground truth data.

Contents

1 Introduction **1**

1.1 Motivation 2

1.2 Research Contributions 4

1.3 Article Contributions 5

2 Literature Review **9**

2.1 Lidar Localisation 9

2.1.1 Classical Methods for Lidar Localisation 10

2.1.2 Deep Learning for Point-Cloud Retrieval 12

2.2 Radar Localisation 13

2.2.1 Classical Methods for Radar State Estimation 14

2.2.2 Deep Learning for Radar State Estimation 14

2.3 Localisation Using Overhead Imagery 15

2.3.1 Cross-View, Intra-Modality 16

2.3.2 Consistent View, Cross-Modality 16

2.4 General Cross-Modality Localisation 17

2.5 Learning Image-to-Image Translation 18

2.6 Learning to Solve Registration 19

2.6.1 Image-Based Registration 19

2.6.2 Points-Based Registration 20

2.7 Sequence Matching for Place Recognition 20

3 RSL-Net: Localising in Satellite Images From a Radar on the Ground **23**

4 Self-Supervised Learning for Using Overhead Imagery as Maps in Outdoor Range Sensor Localization **33**

5 Point-based Metric and Topological Localisation between Lidar and Overhead Imagery **53**

6 Self-Supervised Lidar Place Recognition in Overhead Imagery Using Unpaired Data **79**

7 Concluding Remarks **93**

Appendices

A Preparing Overhead Imagery Data **99**

B Self-Supervised Localisation between Range Sensors and Overhead Imagery **101**

C Get to the Point: Learning Lidar Place Recognition and Metric Localisation Using Overhead Imagery **113**

References **125**

1 Introduction

Localisation, i.e. the ability to autonomously answer the question "where am I?", is of utmost importance in mobile robotics applications, such as autonomous driving and augmented reality, that require an accurate understanding of the surrounding environment. This thesis focuses on large-scale, outdoor mobile robotics scenarios. While an on-board GPS can sometimes answer the question, GPS signals may be inaccurate or unavailable in places like urban canyons. Besides, *redundancy* is extremely necessary for any safety-critical systems. Indeed, a major focus of the robotics community for the past two to three decades has been solving localisation with on-board sensors such as cameras, lidars, and radars.

While vision-only autonomy has shown a certain level of success in the assisted driving industry, active range sensors, such as lidars and Frequency-Modulated Continuous-Wave (FMCW) spinning radars, are still largely considered essential for large-scale safety-critical mobile robotics systems. Lidars and radars are invariant to lighting conditions and have ranges typically longer than 100m with a 360° field-of-view (FOV). Moreover, lidars are more robust against weather conditions than cameras, while FMCW radars are invariant against the weather.

In the typical scenario, a *mapping* phase generally happens before localisation, where sensor data are collected – usually with the same sensor suite that will be used for localisation on the map – as the vehicle travels along the operating

environment. In a later *localisation* phase, the vehicle's pose can be estimated by comparing live sensor data against the prior map, often combining other sources of pose information such as solutions from ego-motion estimation. Here, we follow the topometric paradigm, which distinguishes between two localisation phases, namely *topological* and *metric*. Topological localisation, also known as place recognition, seeks to solve for a robot's coarse pose estimate in an environment, often apropos no prior information. Topological localisation seeks to answer the problem of "where am I?" to perhaps the nearest street or intersection to the robot's location, but often does not provide metre or sub-metre level precision. On the other hand, metric localisation takes the coarse pose estimate from place recognition as an initial starting point and refines the robot's metric pose in $SE(2)$ or $SE(3)$, often using registration techniques such as a point-cloud alignment. This thesis targets both the metric and topological localisation of range sensors, although using overhead imagery rather than prior sensory data as the map database. Finally, some may consider odometry, or ego-motion estimation, as a special case of ego-centric metric localisation, though odometry alone cannot solve for the robot's global pose in the map reference frame.

1.1 Motivation

The majority of existing localisation techniques tend to require a previous mapping phase, where the map data are used to localise the robot given live sensory data. Different from existing techniques that require prior mapping, this thesis presents research on using off-the-shelf, publicly available overhead imagery such as Google[1] or Bing[2] satellite imagery for localising a ground range sensor. Radar or lidar data, when projected to the x-y plane and represented as a bird's-eye view image, share the same overhead perspective as satellite imagery, and capture some geometric cues also observable in satellite imagery which are potentially useful for localisation. Figure 1.1 shows an example of a satellite image and a radar image taken by a Navtech spinning radar at the centre of the satellite image.

[1] `https://developers.google.com/maps/documentation/maps-static/intro/`
[2] `https://docs.microsoft.com/en-us/bingmaps/`

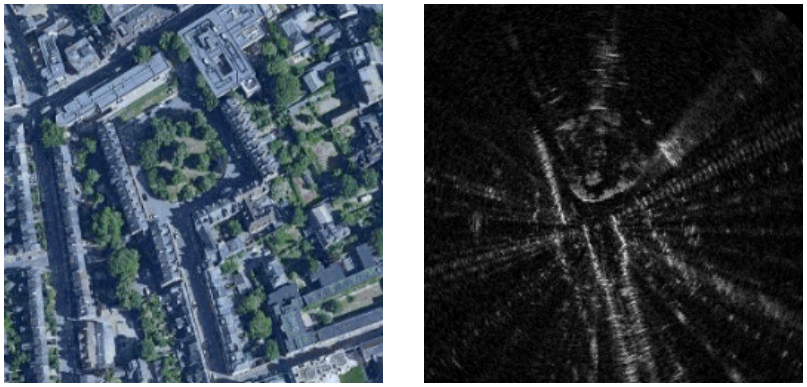


Figure 1.1: A satellite image and a radar image taken at the centre of the satellite image, pose-aligned for visualisation. Many important geometric cues such as edges of buildings and bushes are captured by both the satellite and the radar data, which are potentially useful for cross-comparison.

Using overhead imagery as the "map" for localisation is motivated by the following considerations:

Coverage If the robot travels to a place it has not visited before, then a prior sensor map is entirely unavailable. However, satellite images are readily available almost everywhere in the world, and can easily be obtained either online or stored ahead of time in a map database. While odometry and simultaneous localisation and mapping (SLAM) can provide the ego-motion, satellite imagery in this case can act as a powerful tool for localisation.

Information Redundancy Autonomous navigation is often a safety-critical system. It is therefore desirable to have multiple sensor suites, each providing its own location information, to achieve redundancy. Autonomous robotic platforms operating in real-world applications, such as driver-less vehicles or automated delivery, typically require a combination of sensors, including GPS-INS, cameras, ranges sensors, and inertial measurement units (IMUs). Importantly, public overhead images are prevalent, and usually freely available, adding very little cost and labour to acquire. The localisation solution obtained from overhead imagery can therefore act as an additional information source to further add redundancy.

Map Degradation The prior sensor map might become outdated due to scene changes. Although range sensors are robust against seasonal changes and changes in lighting and appearance, they too can become obsolete due to scene changes such as the addition or demolition of a building. If satellite imagery freely available on the internet is more up-to-date than the sensor map, then satellite images can be used as an alternative source for localisation, as a fall-back choice. Furthermore, in most cases, it is much cheaper to sample satellite images than to reconstruct an up-to-date lidar or radar map.

Global-Frame Place Recognition An on-board GPS, though usually not accurate enough to provide sub-metre level accuracy, can right away report a vehicle's current longitude and latitude information. An autonomous robot can, for example, use this information to locate a sub-region of a city-wise HD map, where their precise pose can be found from metric localisation. In GPS-denied places, localisation against satellite imagery can be used as an alternative for global-frame place recognition, as each satellite image tile is centred at a known longitude and latitude.

1.2 Research Contributions

Localisation between ground range sensors and overhead imagery is challenging, largely unsolved, and under-explored. The greatest challenges arise from the *modality gap*: range sensors and overhead imagery are drastically different data with entirely different sensing physics, the former being range returns (e.g., nanometre-wave for Velodyne lidars and millimetre-wave for Navtech radars) collected on the ground, while the latter is RGB data from overhead. Even when represented as a bird's-eye view scan image, both radar and lidar images are single-channel data where the majority of pixels are dark, and only a small fraction of pixels have range returns, corresponding to regions of the scene that reflect range sensor waves, such as edges of buildings or tree stems. The challenge is perhaps further increased by the complex sensing physics of FMCW radars, which can naturally feature multipath reflections (Figure 1.2), and "ghost" returns in regions without a physical

object [1]. For these reasons, in most cases, range sensor data cannot be directly compared against satellite imagery. While perhaps hand-crafted points-of-interest may be selected from satellite and range sensor images, there is no guarantee they will be consistent across different scenes.

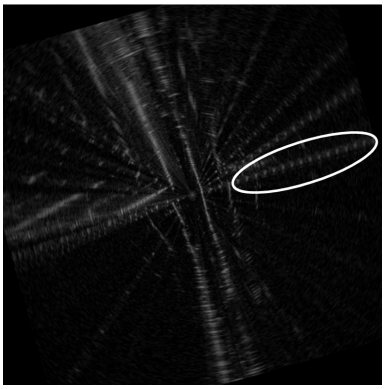


Figure 1.2: An example of multipath reflection in radar signal (circled).

To this end, this thesis makes the following contributions:

- To the best of our knowledge, one of the first methods that learn the metric localisation, or image registration, between a range sensor image and a matched satellite image, from a coarse initial pose estimate from place recognition
- A self-supervised learning strategy where such metric localisation can be learned from noisy training data without metre-level accuracy ground truth
- To the best of our knowledge, one of the first methods to learn both the place recognition and metric localisation of ground range sensors in overhead imagery
- A self-supervised learning strategy where range sensor place recognition using overhead imagery can be learned from unpaired data, relaxing the need for pose ground truth

1.3 Article Contributions

This thesis is built from the following articles listed in chronological order that have been published, accepted, or in preparation to be submitted for publication.

Article **A.1** proposes one of the first methods to learn the metric localisation between a ground radar against satellite imagery. Articles **A.2** and **A.4** then show the cross-modality metric localisation between range sensor data and overhead imagery can be learned in a self-supervised fashion without accurate metric ground truth. Articles **A.3** and **A.5** propose a method that jointly solves both the metric localisation and place recognition of range sensor scans in overhead imagery. Finally, article **A.6** asks whether it is possible to learn the cross-modality place recognition in a self-supervised way without any paired data.

A.1 "RSL-Net: Localising in Satellite Images From a Radar on the Ground" by Tim Y. Tang, Daniele De Martini, Dan Barnes, and Paul Newman was published as a letter in the IEEE Robotics and Automation Letters in January 2020. This work also forms Chapter 3 of this thesis.

A.2 "Self-Supervised Localisation between Range Sensors and Overhead Imagery" by Tim Y. Tang, Daniele De Martini, Shangzhe Wu, and Paul Newman was published and presented as a conference proceeding at Robotics: Science and System XVI in July 2020. This work is included in appendix B of this thesis for completeness.

A.3 "Get to the Point: Learning Lidar Place Recognition and Metric Localisation Using Overhead Imagery" by Tim Y. Tang, Daniele De Martini, and Paul Newman was published and presented as a conference proceeding at Robotics: Science and System XVII in July 2021. This work is included in appendix C of this thesis for completeness.

A.4 "Self-Supervised Learning for Using Overhead Imagery as Maps in Outdoor Range Sensor Localization" by Tim Y. Tang, Daniele De Martini, Shangzhe Wu, and Paul Newman was published as a journal article at The International Journal of Robotics Research in December 2021. This work also forms Chapter 4 of this thesis.

A.5 "Point-based Metric and Topological Localisation between Lidar and Overhead Imagery" by Tim Y. Tang, Daniele De Martini, and Paul Newman was submitted as a journal article to Autonomous Robots, and has been fully accepted in January 2023 for publication. This work also forms Chapter 5 of this thesis.

A.6 "Self-Supervised Lidar Place Recognition in Overhead Imagery Using Unpaired Data" by Tim Y. Tang, Daniele De Martini, and Paul Newman is an ongoing work and in preparation to be submitted as a conference proceeding to Robotics: Science and System XIV for peer review. This work also forms Chapter 6 of this thesis.

2
Literature Review

In this chapter, we review background work related to this thesis, including prior methods for lidar and radar localisation, and other methods that seek to use overhead imagery or other sources of publicly available data as the map for localisation. We also address prior methods related to, or used by, certain modules of our work, including image-to-image translation, and machine learning for solving registration.

2.1 Lidar Localisation

Lidar localisation has received significant attention within the community and is widely considered a well-established area. Here we provide a brief overview of existing work on lidar-based place recognition and metric localisation. Earlier methods are mostly *classical*, using hand-crafted features or key-points combined with classical state estimators. Recent works largely exploit the rapid advancement of *deep learning*, where neural networks consume range sensor data before either learning the localisation end-to-end or feeding to a downstream network-free module. Research presented in this thesis differs from existing work in that we seek to solve lidar localisation against overhead imagery, rather than lidar-to-lidar localisation.

2.1.1 Classical Methods for Lidar Localisation

Place Recognition

The method in [2] divides space occupied by lidar scans into a grid of cells, where a normal distributions transform (NDT) surface representation is created for each cell based on the mean vector and covariance matrix of points within it. Bosse and Zlot [3] designed a regional 3D key-point descriptor based on the eigenvalues, planarity, and cylindrical-ness of point-clouds. Their follow-up work in [4] extracts Gestalt key-points, and proposes a voting approach that solves point-cloud retrieval in sub-linear time. Steder et al. [5] extracted NARF [6] point features and proposed a bag of words (BoW) descriptor for place recognition. M2DP [7] proposes a global descriptor by projecting 3D lidar point-clouds to multiple 2D planes and generating a density descriptor for each plane. Guo et al. [8] proposed a descriptor based on ColorSHOT [9] applied to lidar intensity images for place recognition. SegMatch [10] first clusters point-clouds into discrete segments and extracts descriptors based on eigenvalues and the ensemble of shape histograms per cluster for place recognition. Scan Context [11] considers the top-down view of a lidar scan and encodes the point-cloud into a matrix describing egocentric 2.5D structural information. The authors later proposed Scan Context++ in a follow-up work [12], which is a point-cloud descriptor invariant to both rotation and translation. BVMatch [13] is a recent method that constructs a rotation-invariant descriptor using Log-Gabor filters applied to bird's-eye view lidar images in polar representation.

Metric Localisation

Early work in [14] proposes a GraphSLAM objective function that integrates GPS data, IMU, wheel odometry, and lidar data. The error term for lidar measurements computes the difference between the observed infrared reflectivity and the expected infrared reflectivity given a measurement model and a map. This was extended in [15], where the map is represented probabilistically as Gaussian distributions of remittance values, but this method still relies on laser reflectivity. Later methods make use of point coordinates rather than laser reflectivity in lidar localisation, where

point-cloud registration can be solved using methods such as iterative closest point (ICP) [16] or NDT [17, 18]. Newman et al. [19, 20] generated local swathes from a 2D push-broom lidar to localise in a prior 3D point-cloud map. The point-cloud alignment in $SE(2)$ is achieved by minimising a cost function that measures the Kullback-Leibler (KL)-divergence between the local swathe and the prior point-cloud. Barfoot et al. [21] made use of lidar intensity images, where SURF [22] features extracted from intensity images are used in a Visual Teach and Repeat (VT&R) pipeline with continuous-time pose estimation. Monte Carlo Localisation (MCL) was widely applied in lidar localisation [14, 23–26] as an alternative to computing a single registration per scan. Within these methods, [24] builds a lidar sensor model based on beam endpoints, while the methods in [25, 26] discretize the world into 2-D x-y grids and represent lidar data as a map of Gaussian mixtures describing the z value distribution. The registration between a live point-cloud and the Gaussian mixture map is found by exhaustive search. The method in [27] detects line features from lidar reflectivity and store them as an occupancy grid map, and localisation is performed by correlation matching. Using a map represented as a triangular mesh, the method in [28] renders a synthetic range image from the mesh map and compares it against the live range image from lidar in an MCL framework. Recent work in [29] proposes a direct method for registering against a prior lidar map that does not require features or point correspondences using a non-linear optimisation.

In addition to map-based localisation, lidar SLAM is also an extensively researched area. G-ICP [30] combines standard ICP and point-to-plane ICP for scan registration. LOAM [31] extracts salient edge points as features and uses a constant-velocity motion model to project points within a sweep to the same reference frame to handle distortion. The LOAM framework uses a higher frequency odometry stream for quick ego-motion update, and a lower frequency mapping stream for pose refinement. IMLS-SLAM [32] stores previous localised scans using the Implicit Moving Least Squares (IMLS) surface representation, and relies on a scan-to-model matching similar to [33] rather than the classical scan-to-scan matching for registration. SuMa [34] aggregates a surfel-based map, and loop

closure detection is handled by registering the current point-cloud to a rendered view from the surfel map. SuMa++ [35] integrates semantic prediction into the SuMa [34] pipeline. The semantic map allows for filtering out dynamic objects, and introducing additional semantic constraints into ICP, leading to higher metric accuracy. The method in [36] proposes a factor graph that directly optimises a global matching cost between frames over the entire map, different from standard pose-graph approaches. CT-ICP [37] uses a continuous-time variant of ICP to handle point-cloud motion distortion. To detect loop closures, the method in [37] projects lidar data to a 2D elevation image, where rotation-invariant features are detected.

2.1.2 Deep Learning for Point-Cloud Retrieval

Learning-based methods for lidar (and other point-cloud type data) place recognition can be categorised by how the networks consume the input data. With the recent advancements of point-based neural networks such as PointNet [38], a lidar point-cloud may directly be fed as input to a deep network. Other methods project the lidar data to images or voxels before applying 2D or 3D convolutions. In this section, we briefly outline existing learning-based methods for solving retrieval, while learning-based methods for solving point-cloud registration are reviewed in Section 2.6.2.

Point-Based Methods

PointNet [38] is a pioneering work that allows neural networks to directly operate on 3D point coordinates through shared multilayer perceptrons (MLPs) applied to each point. For point-cloud retrieval, the approach is to typically first learn per-point local features using a point-based backbone such as PointNet, and then aggregate them into a global per-point-cloud descriptor by pooling [39], normalisation [40], learned layers [39, 41–43] such as a NetVLAD [44] layer, or BoW [45].

Specifically, PointNetVLAD [41] applies NetVLAD after a PointNet backbone for learning point-cloud descriptors. Rather than operating on each point individually, LPD-Net [39] conducts neighbourhood aggregation on the local features via a graph neural network before feeding to NetVLAD, and is shown to outperform

PointNetVLAD. PCAN [42] utilises attention to predict the significance of each feature's relevance, and selects task-specific features for global descriptor construction. DH3D [43] first optimises local features for point-matching using ground truth correspondences, before aggregating to a global descriptor with NetVLAD. Retriever [46] is a recent method that uses a KPConv-based [47] network to compress the point-cloud first, followed by a perceiver-based [48] architecture for descriptor learning.

Projection Methods

Sun et al. [49] expressed lidar data as bird's-eye view images, which are used to directly regress the robot's global pose, similar in spirit to PoseNet [50]. Here the "map" is in practice memorised by the neural network, and this approach would require place-dependent re-training. Yin et al. [51] proposed a Generative Adversarial Network (GAN) training to learn features from lidar bird's-eye view images for place recognition. MinkLoc3D [52] quantised point-clouds to sparse single-channel 3D voxels, where values are set to 1 for non-empty voxels. 3D convolutions are then applied to the voxels to learn local features, which are aggregated to a global descriptor via Generalised-Mean pooling [53]. This work was extended in [54] for voxel-based learning on spherical representations of lidar data, while also using intensity values. PVT3D [55] combines two branches of networks, one that operates on points with a PointNet backbone and another that operates on quantised voxels. OverlapNet [56] is a recent method that extracts 2D range, normal, intensity, and semantic images from lidar, and applies 2D convolutions to regress an overlap between two lidar scans for detecting loop closure, achieving state-of-the-art accuracy on benchmarks. Follow-up works include OverlapTransformer [57], which learns a rotation-invariant descriptor from a Transformer-based architecture [58] from lidar range images, and SeqOT [59], which handles sequential data.

2.2 Radar Localisation

Spinning FMCW radar manufactured by Navtech has been an increasingly popular sensor in the robotics community, leading to the creation of multiple large-scale

datasets [60–63]. Though having no height information, FMCW radars have a longer sensing range than typical lidars and are invariant to weather conditions due to their wavelengths. However, radar signals are complex and difficult to process compared to lidars, as shown in Chapter 1. Classical methods for radar-based state estimation heavily targeted odometry and a number of them on SLAM with loop closures, while learning-based methods were proposed for both registration and place recognition. Research presented in this thesis differs from existing works in that we seek to solve radar localisation against overhead imagery, rather than against a map of radar data.

2.2.1 Classical Methods for Radar State Estimation

Pioneering radar odometry works by Cen et al. [1, 64] propose detecting key-points from signal smoothing and utilise a data association method based on pairwise distance compatibility. Some of the other recent works on radar odometry propose detecting key-points by thresholding and using NDT matching for registration [65] or conservative filtering for points extraction [66]. RadarSLAM [67] extracts SURF key-points and features from radar images, and applies M2DP [7] to build a descriptor for detecting loop closures. Rather than extracting key-points as in [1, 64–67], PhaRaO [68] solves odometry directly using radar images via phase correlation. The work by Burnett et al. [69] studies the impact of motion distortion and Doppler effects in radar motion estimation. The work in [70] presents a method that leverages a constant-curvature vehicle motion constraint to refine data association in point-based radar odometry. ORORA [71] proposes to decouple radar motion estimation into a graduated non-convexity-based rotation estimation and an anisotropic component-wise translation estimation, for outlier-robust odometry. Recent work in [72] presents a de-noising method to remove multipath reflections and "ghost" returns in radar signal, and uses NDT for odometry and localisation.

2.2.2 Deep Learning for Radar State Estimation

Aldera et al. [73] used a focus of attention policy to learn to down-sample input radar data for fast odometry. UnderTheRadar [74] learns to predict key-points

from radar images, where each key-point (corresponding to a sub-pixel location) is associated with a local descriptor used for soft data association. Odometry is solved by applying singular value decomposition (SVD) on the matched pairs of key-points. While the method in [74] is trained using ground truth odometry, [75] proposes an Expectation-Maximisation approach combined with probabilistic estimation to learn key-points and features in an unsupervised fashion. State-of-the-art learned radar odometry in [76] learns to mask regions of the scene that are distracting for odometry, such as moving vehicles. The $SE(2)$ pose estimation is performed by a correlation-maximisation approach on the masked radar images that exhaustively searches for the best x, y, θ offset between consecutive frames. This was extended in [77] where the search in rotation and translation is decoupled, greatly speeding up the computation.

Saftescu et al. [78] trained a Convolutional Neural Network (CNN) followed by a NetVLAD layer on radar images in polar representation with circular padding, and learned a rotation-invariant descriptor for retrieval. This was extended for sequence-based radar place recognition in [79] and an end-to-end, topological and metric localisation pipeline in [80]. The method in [81] proposes contrastive learning for radar place recognition rather than using triplets as in [78–80]. Instead of using a NetVLAD layer, the work in [82] applies Generalised-Mean pooling to aggregate learned CNN features to a global descriptor.

2.3 Localisation Using Overhead Imagery

The use of overhead imagery for solving the localisation problem has been explored in the past for either ground camera data or ground range sensor data. In the computer vision community, the problem of *geo-localisation* seeks to solve the place recognition of a ground camera image using an overhead image database. In this case, the modality between live data and map data is the same as both are RGB images. However, the two data types have different perspectives, with the former being forward-facing images taken on the ground and the latter being bird's-eye view images taken from above. When range sensor data are projected

to bird's-eye view for localisation in overhead imagery, the perspectives become consistent. However, in this case, the data modalities differ as range sensor data are compared to RGB images.

2.3.1 Cross-View, Intra-Modality

Early work by Lin et al. [83] uses hand-crafted features based on histograms of oriented gradients (HoG), self-similarity, gist, and colour histograms, to train a support vector machine (SVM) classifier to perform cross-view geolocalisation. Hu et al. [84] proposed CVM-Net, using two streams of CNN + NetVLAD Siamese networks to learn cross-view retrieval. Li et al. tackled geolocalisation by supplying orientation maps [85], predicting cross-view orientation [86], learning spatial attention [87], and using optimal feature transport [88].

2.3.2 Consistent View, Cross-Modality

Localising a point-cloud against overhead imagery is a type of consistent view, cross-modality localisation. Early work in this research theme mostly relies on hand-crafted methods to bridge the domain gap between point data and overhead imagery. The method in [89] extracted binary images of edges from overhead images before comparing them to bird's-eye view point-cloud images. Veronese et al. [90] accumulated lidar scans to form dense lidar intensity images, and matched them against satellite images utilising Normalised Mutual Information. Other methods of overhead imagery pre-processing before comparing to point-cloud data include using edge-detection [91] or semantic segmentation [92]. Hussein et al. [93] localised in a forest environment by detecting tree stems with lidar and comparing them against tree crowns in overhead imagery. The work in this thesis differs from these methods in that we seek to learn to bridge the modality gap rather than relying on hand-crafted features specific to a fixed environment.

Several recent methods concurrent with this thesis have also proposed to apply deep learning for cross-comparison between ground range sensors and overhead imagery. Zhu et al. [94] learned a matching probability between lidar grid maps and

overhead imagery, which are used as factors in a pose graph optimisation to reduce drift in a lidar SLAM pipeline. In a similar style but under an MCL framework, Fu et al. [95] learned a scalar weight between the height and intensity data of a live lidar scan and a satellite image queried at the position of a particle, to update the weight of the particles. Fervers et al. [96] combined ground camera images and bird's-eye view lidar images to solve the registration against overhead imagery. These methods differ from our work in combining range sensor data with RGB images or using overhead imagery as an auxiliary signal for pose factors in a SLAM optimisation. In contrast, this thesis seeks to target stand-alone range sensor localisation against overhead imagery without relying on other sensors or pose estimation pipelines.

Most recently, Kim et al. [97] trained a semantic segmentation network using hand-annotated satellite images, and extracted building outlines while also performing orthorectification. This practically converted satellite imagery to a representation similar to building outlines in OpenStreetMap (OSM). Localisation is then solved by computing mutual information between lidar scans against building outlines.

2.4 General Cross-Modality Localisation

Other forms of cross-modality localisation, where the on-board sensor data differ from the map data, have been previously explored under various set-ups and scenarios. Among them are methods that seek to localise a ground camera in a prior lidar map [98–100]. In such applications, the prior lidar maps are created by accumulating all lidar scans to the same reference frame through a mapping phase, and therefore have appearances similar to, or at least comparable against visual data, rather than individual scans of lidar which are sparse and drastically different from overhead images. The method in [101] solves the global localisation of a lidar against orbital elevation maps by detecting features from topographic peaks. For indoor navigation, it is possible to localise an on-board laser scanner or stereo camera in a map of architectural floor plans [102–104]. Iannucci et al. [105] localised an automotive radar in a camera map, by matching radar point

returns against SIFT [106] features extracted from camera images. Yin et al. [107] used a GAN style transfer to generate synthetic lidar images from radar data, and achieved radar-to-lidar metric localisation. The method in [108] combines style transfer with joint feature space learning to find cross-modal retrieval between RGB and infra-red images. Notably, the methods in [107, 108] can learn cross-modal localisation or retrieval without paired data.

OSM is another freely available public resource that can be used as a powerful tool for outdoor localisation. Without directly using sensor data, concurrent works in [109, 110] first computed odometry, and then matched ego-motion trajectory estimations against road layouts extracted from OSM for localisation. Vysotska and Stachniss [111] extracted building contours from OSM. Error terms are constructed based on the alignment between building contours and projected lidar returns in a pose graph-based SLAM. The methods in [112, 113] exploit the semantic labels in OSM data and extracted semantic information from lidar data. The cross-modality comparison is achieved by forming a semantic descriptor consistent across lidar and OSM. Recent work by Cho et al. [114] proposes a hand-crafted descriptor based on the distance to the nearest building in each azimuth consistent between lidar and OSM data.

Other than state estimation, robot path planning can also be achieved using overhead imagery [115, 116] or OSM [117].

2.5 Learning Image-to-Image Translation

This thesis's work uses variations of image-to-image translation to bridge the modality gap between range sensor and overhead images. Specifically, overhead images are used to generate synthetic range sensor images in Chapters 3, 4 and 6, and grey-scale occupancy images in Chapter 5. Here we provide a brief overview of the recent advancement of image-to-image translation from the computer vision community.

Pioneering work on GAN [118] generates synthetic images from random noise vectors by learning in an adversarial fashion a generator network, and a discriminator

network that classifies images as real or fake. Pix2Pix [119] learns to transfer an image from one style to another using conditional adversarial networks. While Pix2Pix requires paired data for training, unpaired image-to-image translation can be achieved with cycle consistency [120] between the input and the reconstructed image, the assumption of a shared latent space [121, 122], or by contrastive learning [123]. Several prior works were proposed to learn multi-modal image generation where the output modality can be controlled [122, 124, 125]. Finally, related to the work in this thesis, in particular Chapter 4, some methods are *geometry-aware*, using separate encoder and/or decoder networks to disentangle geometry from appearance. The methods in [126–128] can interpolate the geometry and appearance of the output image at inference time.

2.6 Learning to Solve Registration

The works in Chapters 3 to 5 seek to learn the metric localisation, or registration between range sensor and overhead data, where the networks can be trained end-to-end with a loss on pose only. Prior work was proposed to apply deep learning for solving image registration or point-cloud alignment, which can be applied to robot metric localisation.

2.6.1 Image-Based Registration

Barsan et al. [129] represented lidar data as intensity images, and learned a deep embedding for image registration. To align two images, the $SE(2)$ offset between their deep embeddings is found by an exhaustive search in x, y, θ, which can be performed efficiently in the Fourier domain by maximising correlation. The correlation maximisation module is parameter-free and fully differentiable, allowing upstream networks to be trained end-to-end with only a loss on the pose. This was applied to radar images for learned radar odometry in [76] and [77], where [77] decouples translation and rotation to achieve faster computation time.

2.6.2 Points-Based Registration

It is well known that the convergence of ICP [130] is sensitive to the initial alignment. 3DFeatNet [131] learns local descriptors with PointNet++ [132], trained with a triplet loss. Deep Closest Point (DCP) proposes to learn point-wise features using a DGCNN [133] backbone combined with a Transformer module to learn point-wise features. Soft point-wise data association can be established by feature matching, where SVD is applied to soft point correspondences for solving $SE(3)$ registration. DCP is trained directly with a loss on pose rather than triplet loss for descriptor correspondences as in [131]. Notably, DCP was shown to lead to better convergence than ICP at larger initial offsets. PRNet [134] builds upon DCP targeting partial-to-partial registration, and uses Gumbel-Softmax for hard point correspondences and iterative alignment rather than one-shot SVD. DeepGMR [135] represents a point-cloud as Gaussian mixtures, and learns point-to-distribution rather than point-to-point correspondences, greatly reducing computation cost for large point-clouds. Alternative to point-based networks, 3D convolutions were applied to learn descriptors from RGB-D reconstructions for matching in [136].

2.7 Sequence Matching for Place Recognition

The work presented in 6 seeks to tackle cross-modality place recognition from unpaired data utilising sequence alignment. A number of prior works have relied on sequence matching to solve place recognition, but usually for data within the same sensor suite (e.g., visual place recognition), albeit under appearance and scene changes due to different weather, seasons, and lighting conditions. Given a frame of live sensor data, rather than finding the global match against all prior data, sequence-based methods search for local best matches resulting from the alignment.

Early work by Ho and Newman [137] extracts SIFT descriptors from images, and calculated the cosine distance between all possible pairs of descriptors to construct a similarity matrix. A modified Smith-Waterman algorithm [138] was applied on the similarity matrix for sequence alignment, in order to detect loop

closures. To enhance local contrast for better sequence matching, the method in [137] proposes removing eigenvectors from the similarity matrix to form a rank-reduced one before alignment. Liu and Zhang [139] built a similarity matrix using Gabor-Gist descriptors, and applied Principal Component Analysis (PCA) to reduce its dimensionality for increasing contrast. SeqSLAM [140] uses a sequence of visual data for place recognition across changing seasons, weather, and lighting conditions. In particular, SeqSLAM performs a local 1D patch normalisation on the difference matrix to enhance its contrast, then finds a diagonal line from the top left to the right-most column that minimises the difference, by searching through various velocities. Fast-SeqSLAM [141] extends SeqSLAM to be more computationally efficient, by only computing the distance to the nearest N neighbours rather than exhaustively over all possible pairs, and utilising a sparse difference matrix. Rather than relying on pre-defined constant-velocity search paths as in SeqSLAM, the method in [142] models sequence alignment through the similarity matrix using a Hidden Markov Model, and finds the most probable path using the Viterbi algorithm. The method in [143] builds the similarity matrix using pre-trained OverFeat [144] features, and models the online sequence alignment problem as finding the shortest path in a graph, where nodes represent potential matches and edges represent the transition between potential matches. The nodes are expanded based on a heuristic for fast searching. The authors' follow-up work [145, 146] handles localisation against a map of multiple sequences, where edges now allow for transition between reference sequences, as well as relocalisation after losing track. The capability to utilise multiple sequences in [145] allows the use of Google Street View images as a sequence. The method in [147] proposes K Standardisation of Descriptors, and also applies PCA for descriptor dimensionality reduction to enhance local contrast in the similarity matrix, similar to [139]. SeqNet [148] learns a sequential descriptor from a sequence of images and single descriptors from individual images. Given a sequence of query images, the sequential descriptor is used to find the top K sequences of reference image candidates. Each candidate is then compared to the query sequence using single descriptors without velocity searching as in SeqSLAM

[140]. The method in [79] applies the constant-velocity search paths proposed in SeqSLAM for sequence-based radar localisation.

3

RSL-Net: Localising in Satellite Images From a Radar on the Ground

This article was published as a letter in the IEEE Robotics and Automation Letters in January 2020, and presented at IEEE International Conference on Robotics and Automation 2020.

Background and Motivation

Most existing methods for metric localisation between range sensors and overhead imagery tend to rely on hand-crafted techniques to pre-process the data, where features observable in both data modalities are kept for comparison. Hand-crafted features are usually designed for a specific type of environment such as campus [91] or forest [93], and usually cannot generalise well to different places. In this chapter, we present, to the best of our knowledge, one of the first methods to learn the registration between a ground radar image against satellite images, free from using hand-picked features.

RSL-Net: Localising in Satellite Images From a Radar on the Ground

Tim Yuqing Tang, Daniele De Martini, Dan Barnes, and Paul Newman

***Abstract*—This letter is about localising a vehicle in an overhead image using FMCW radar mounted on a ground vehicle. FMCW radar offers extraordinary promise and efficacy for vehicle localisation. It is impervious to all weather types and lighting conditions. However the complexity of the interactions between millimetre radar wave and the physical environment makes it a challenging domain. Infrastructure-free large-scale radar-based localisation is in its infancy. Typically here a map is built and suitable techniques, compatible with the nature of sensor, are brought to bear. In this work we eschew the need for a radar-based map; instead we simply use an overhead image – a resource readily available everywhere. This letter introduces a method that not only naturally deals with the complexity of the signal type but does so in the context of cross modal processing.**

***Index Terms*—Autonomous vehicle navigation, deep learning in robotics and automation, localization, range sensing.**

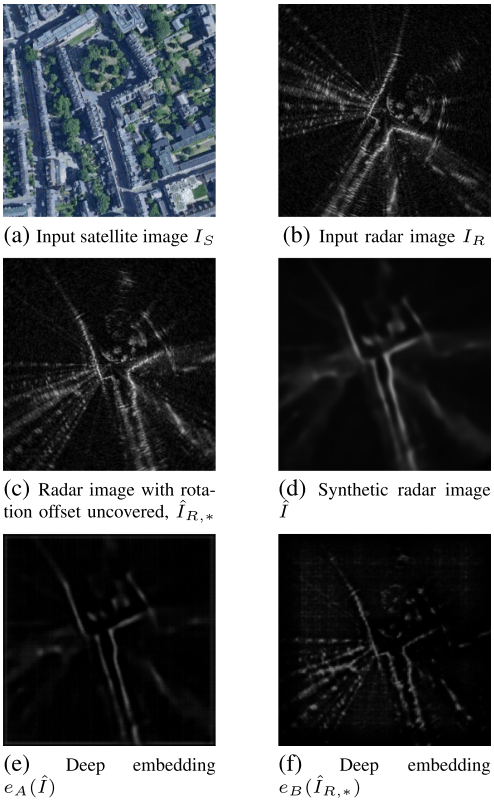
I. Introduction

FOR autonomous vehicles to operate reliably they must be able to localise with respect to their surrounding environment. While localisation in benign environments is generally a well-studied concern, long-term localisation in complex environments under changing weather and scene conditions remains an open problem. Localisation using vision sensors can suffer due to a different lighting condition as in the map, while, although robust against lighting conditions, lidars are sensitive to foul weather such as rain, fog, and snow.

Current approaches typically involve mapping the environment multiple times using on-board sensors across different scene and appearance conditions [1], [2], or learning an appearance transfer across domains, such as daytime and night images [3]. For scenarios of unusable or missing prior sensor maps, publicly available overhead imagery offers a cheap, easily accessible fallback solution.

To this end, we propose Radar-Satellite Localisation Network (RSL-Net), which localises an on-board Frequency- Modulated Continuous-Wave (FMCW) scanning radar against overhead imagery. Scanning radar sensors are invariant to lighting and weather, making them ideal for operating outdoor, and radar images can be displayed in a "birds-eye" view similar to overhead imagery, making them suitable for the task. Though we make use of Google satellite imagery in this work, RSL-Net is not limited to satellite images and can be trained for any types of overhead imagery.

Given as inputs an overhead image and a radar image with an unknown $SE(2)$ pose offset and sufficient overlap, RSL-Net generates a synthetic image that preserves the appearance of the input radar image and the pose of the input overhead image. Shown in Fig. 1, this is done by first uncovering the rotation offset between the two (c), and then generating the synthetic image (d). The synthetic image and radar image are then embedded into a joint domain (e and f), where the 2D position offset is found by an exhaustive search in the (x, y) solution space, thus



(a) Input satellite image I_S (b) Input radar image I_R (c) Radar image with rotation offset uncovered, $\hat{I}_{R,*}$ (d) Synthetic radar image $\hat{I}$ (e) Deep embedding $e_A(\hat{I})$ (f) Deep embedding $e_B(\hat{I}_{R,*})$

Fig. 1. Given an input satellite-radar pair I_S and I_R with an unknown offset, RSL-Net first uncovers the rotation offset θ and creates $\hat{I}_{R,*}$, which is I_R rotated to match the heading as in I_S. Taking in I_S and $\hat{I}_{R,*}$ as inputs, a generator network produces $\hat{I}$ conditioned on the appearance of $\hat{I}_{R,*}$ and the pose of I_S. The images $\hat{I}$ and $\hat{I}_{R,*}$ are projected to a joint deep embedding by functions e_A and e_B, where their translation offset in (x, y) is found.

Manuscript received September 10, 2019; accepted December 23, 2019. Date of publication January 13, 2020; date of current version January 30, 2020. This letter was recommended for publication by Associate Editor V. Indelman and Editor S. Behnke upon evaluation of the reviewers' comments. The work of T. Y. Tang was supported by the Postgraduate Scholarship - Doctoral Program (PGS-D) from the Natural Sciences and Engineering Research Council of Canada. The work of D. Barnes was supported by UK EPSRC Doctoral Training Partnership. The work of D. De Martini and P. Newman were supported by EPSRC Programme Grant EP/M019918/1. *(Corresponding author: Tim Yuqing Tang.)*

The authors are with the Oxford Robotics Institute, University of Oxford, Oxford OX1 2JD, U.K. (e-mail: ttang@robots.ox.ac.uk; daniele.demartini@eng.ox.ac.uk; dbarnes@robots.ox.ac.uk; pnewman@robots.ox.ac.uk).

Digital Object Identifier 10.1109/LRA.2020.2965907

2377-3766 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

solving the remaining two degrees of freedom (DoFs) of the $SE(2)$ pose.

We do not seek to solve the general place recognition problem, but rather assume that a very coarse initial pose estimate can be found by other means, for example a combination of radar odometry [4] and noisy Global Navigation Satellite System (GNSS). Once we have a coarse initial pose estimate (10 s of metres error is expected), our method utilises overhead imagery to compute a refined metric localisation.

Despite radars being immune to weather and lighting conditions, localising a ground range sensor against overhead imagery remains challenging for several reasons. Radar images are collections of frequency vs energy tuples (where frequency is a proxy for range), and are therefore intrinsically different than overhead images, making a direct comparison quite daunting. As is true for most on-board sensors, due to occlusions, the radar only observes a limited region of the scene captured by the overhead image. Parts of the scene such as cars, pedestrians, and temporary road signs may exist in the live data but not in overhead imagery, or vice versa.

To the best of our knowledge, this letter presents the first method to learn the metric localisation of an on-board range sensor using overhead imagery. By evaluating across datasets containing both an urban setting and highway driving in rural regions, we show that our method achieves high localisation accuracy in various types of environments. We validate on test data covering approximately 10 min of driving and 10 km of distance travelled.

II. Related Work

A. Localisation Against Overhead Imagery

Prior work for metrically localising a ground sensor against overhead imagery are mostly classical estimation methods. Noda *et al.* [5] localised a ground camera against overhead images using SURF features. The methods in [6]–[8] achieve a similar task using other hand-crafted features. Senlet *et al.* [9] presented a framework for dense matching between ground stereo images and satellite images. The work in [10] localises a ground laser using edge points detected from aerial images. The method in [11] registers lidar intensity images against overhead images by maximising mutual information. By finding features from vertical structures, FLAG [12] achieves air-ground outdoor localisation as well as indoor localisation using floor plans. Forster *et al.* [13] localised a ground laser using dense reconstruction from overhead images taken by an aerial vehicle. These methods rely on hand-crafted systems to work in specific setups, but does not demonstrate generalisation in previously unseen environments.

Learning-based methods have been proposed to solve place recognition using overhead imagery, but not metric localisation. Chu *et al.* [14] learned a feature dictionary to compute the probability that a ground image is taken with a particular pose using overhead imagery. Given a database of satellite images, Kim *et al.* [15] trained a Siamese network to learn matched ground-satellite image pairs, which are used in particle filter localisation. These methods do not learn the sensor pose end-to-end. Similarly, work in ground-to-aerial geolocalisation such as [16]–[18] addresses place recognition by finding ground-satellite image matches, but does not learn metric localisation of the ground sensor.

B. Localisation Using Other Publicly Available Data

Some methods address the metric localisation of a ground sensor using other types of off-the-shelf, publicly available data, particularly OpenStreetMap (OSM). Hentschel *et al.* [19] used building tags in OSM as additional observations to update the weights of particles in particle filter localisation. Brubaker *et al.* [20] built a graph-based representation of OSM and a probabilistic model to match VO paths to the map. Similarly, Floros *et al.* [21] used Chamfer matching to match to VO paths to OSM paths. Methods utilising OSM typically require labels for buildings or path information, while our method relies on overhead imagery and does not depend on labelled information.

C. Deep Learning for Outdoor Localisation

Given training data from a previously seen environment, the work by Kendall *et al.* [22], [23] directly regresses the 6-DoF camera pose for relocalisation. For a live query image, Laskar *et al.* [24] learned a descriptor to find matches from a database of prior images, and regressed the pairwise relative pose. The method presented in [25] solves camera relocalisation by learning 3D scene coordinates. MapNet [26] learns a data-driven map representation for localisation, where geometric constraints used for state estimation are formulated as network losses. Sarlin *et al.* [27] learned keypoints and descriptors for camera localisation. Recent work in [28] addresses localisation using lidar point-clouds by learning keypoint descriptors through 3D convolutions.

Related to our work is the method by Barsan *et al.* [29], where live lidar intensity images and lidar intensity maps are embedded to a joint space for localisation. Inspired by [29], our method also searches for the translational offset between two deep embeddings by convolving them. Our work and the methods in [28] and [29] compute the pose offset by exhaustively searching through the solution space, rather than regressing the pose with a fully connected layer as in [22] and [23]. Existing work relies on prior data gathered with the same sensor modality as the live data, whereas our method is concerned with localising a radar against overhead imagery.

D. Conditional Image Generation

An important part of RSL-Net is a generator network conditioned on an input overhead image and an input radar image. Building upon Pix2Pix [30] which performs image-to-image translation using conditional Generative Adversarial Network (GANs), the work in [31] achieves multi-modal generation where the input image is conditioned on a random latent vector. Similarly, IcGAN [32] edits an input image using a conditional vector. The network in [33] takes in two input images, where the output contains attributes from both inputs, for example shape as in the first input and color as in the second input. The generator used in [34] takes in a person image and a pose image as inputs; the network generates an output person image that preserves the appearance of the input person image and the pose in the pose image. Likewise, the generator we use takes in an overhead image and a radar image as inputs, and outputs a generated image that preserves the appearance of the input radar image, with a pose being that of the overhead image.

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

III. Radar-Satellite Localisation Network

Given a live radar image I_R and a satellite image I_S, sampled at a coarse initial pose estimate (x_0, y_0, θ_0), we seek to estimate the radar's pose by localising against satellite imagery. We stress that the initial pose estimate does not need to be accurate, but only needs to be good enough such that a sufficient portion of the scene captured by the satellite image is also observed by the radar. The scale of the satellite image is known a priori, the only unknown pose parameters are a rotation and a translation, making this an estimation problem in $SE(2)$. I_R and I_S are fed into RSL-Net to compute the $SE(2)$ offset between them parametrised as (x, y, θ), which provides a refined solution to the radar's pose.

A. Overview of the Pipeline

To localise a radar image against a satellite image, our approach first seeks to generate a synthetic radar image from the satellite image, which is used downstream to compute the pose against the radar image itself. Specifically, we wish to learn a function f such that

$$f\colon (I_R, I_S) \to I, \tag{1}$$

where I is a synthetic radar image that preserves the appearance of I_R but has the same pose as I_S. We can obtain targets for supervising the training as we have ground truth pose between I_R and I_S, which we use to transform I_R into I.

We learn $f(I_R, I_S)$ rather than $f(I_S)$, since radar data is available at test time to condition f. We show in Section VI that this leads to improved results than learning $f : I_S \to I$, which does not depend on I_R. However, training f to condition on both I_R and I_S is non-trivial as I_R and I_S are offset by an $SE(2)$ pose. Standard Convolutional Neural Network (CNNs) are equivariant[1] to translation, but not to rotation. Replacing CNN filters with a specific family of filters [35], [36] can allow for equivariance to other forms of transformations, but limits the types of filters the network can learn. Other approaches seek to learn canonical coordinate representations [37], which are not applicable to our task as canonical representations do not exist for arbitrary radar images.

For this reason, rather than tackling the issue with $SE(2)$ equivariance, we break the problem of learning f into two sequential steps. First, given I_R and I_S with an unknown $SE(2)$ offset, we seek to learn a function h that infers the rotation offset between them; this is done by searching through a discretized solution space of rotations. Specifically, we first rotate I_R about its centre by angles with small increments $\{\theta_1, \theta_2, \ldots, \theta_{N_\theta}\}$ to create an ensemble of images $\{I_R\}_{N_\theta} = \{I_{R,1}, I_{R,2}, \ldots, I_{R,N_\theta}\}$, where each $I_{R,n}$ is a rotated version of I_R. We seek to learn

$$h\colon (\{I_R\}_{N_\theta}, I_S) \to I_{R,*}, \tag{2}$$

where $I_{R,*}$ is an image from the ensemble whose heading angle θ most closely matches with that of I_S. $I_{R,*}$ is then offset with I_S only in translation, while the other images in the ensemble $\{I_R\}_{N_\theta}$ are offset with I_S by rotation and translation. As a result, the optimal θ estimate is also inferred in this process.

[1]A mapping $f : \mathcal{X} \to \mathcal{Y}$ is *equivariant* to a group of transformations Φ, if for any $\phi \in \Phi$, there exists a transformation $\psi \in \Psi$ such that $\psi(f(\chi)) = f(\phi(\chi)), \forall \chi \in \mathcal{X}$. *Invariance* is a special case of equivariance, such that f is invariant to Φ if, for any $\phi \in \Phi$, $f(\chi) = f(\phi(\chi))$.

Without any prior estimate on heading, the search space $\{\theta_1, \theta_2, \ldots, \theta_{N_\theta}\}$ should contain rotations ranging from $-\pi$ to π with fine discretization. However, we can limit the range of solution space to search if we have a coarse estimate on heading, as shown in Section IV.

After finding $I_{R,*}$ we learn a function g to generate I :

$$g\colon (I_{R,*}, I_S) \to I. \tag{3}$$

Learning g to condition on $I_{R,*}$ and I_S is simpler than learning f to condition on I_R and I_S, as in (1), since $I_{R,*}$ and I_S are offset only in translation which convolutions are inherently equivariant to. As such, we have replaced learning f with learning $g : (h(\{I_R\}_{N_\theta}, I_S), I_S) \to I$. By pre-discovering the rotation, we utilise a simple yet effective strategy to take full advantage of the input radar image when generating I, without imposing any restrictions on the network filters learned.

I has the same pose as I_S, therefore I and $I_{R,*}$ are offset only by a translation. Finally, we embed I and $I_{R,*}$ to a joint domain to compute their offset in translation:

$$p\colon (e_A(I), e_B(I_{R,*})) \to \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^2, \tag{4}$$

where e_A and e_B are embedding functions to be learned, and p is a pre-defined function that searches for the (x, y) offset between two images by maximising a correlation surface.

Our network is trained in multiple stages. We first pre-train individually a "rotation selector" network for learning h, and a generator for learning g. We then train all modules together end-to-end, including networks for learning the embedding functions e_A and e_B. The various stages of training are detailed in Sections III-B to III-D, while the overall architecture of RSL-Net is shown in Fig. 4.

B. Rotation Selector Network

We design a rotation selector network such that the network learns to select an image $I_{R,*}$ from $\{I_R\}_{N_\theta}$ with a heading angle θ that most closely matches with I_S. To make the network differentiable, as an approximation of (2), the output of the network $\hat{I}_{R,*}$ is learned to be a linear combination of all images from the input ensemble:

$$\hat{I}_{R,*} = h\left(\{I_R\}_{N_\theta}, I_S\right) = \sum_{n=1}^{N_\theta} \alpha_n I_{R,n}$$

$$I_{R,n} \in \{I_R\}_{N_\theta}, \sum_{n=1}^{N_\theta} \alpha_n = 1, \tag{5}$$

where α_n are learned scalar coefficients.

The architecture of the network is shown in Fig. 2: the network consists of 5 down-convolutional layers with stride 2 followed by 6 ResNet blocks [38], where we sum along the H, W, C dimensions to arrive at a vector $\begin{bmatrix}\alpha_1 & \cdots & \alpha_{N_\theta}\end{bmatrix}^T \in \mathbb{R}^{N_\theta}$. We apply a softmax such that all elements of the vector sum to 1.

The pre-training is supervised using the ground truth image $I_{R,*}$, obtained by rotating I_R to match the heading as in I_S, but maintaining the translational offset. We use an L1 loss between the output image and the target image:

$$\mathcal{L}_h = \left\| h\left(\{I_R\}_{N_\theta}, I_S\right) - I_{R,*} \right\|_1. \tag{6}$$

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

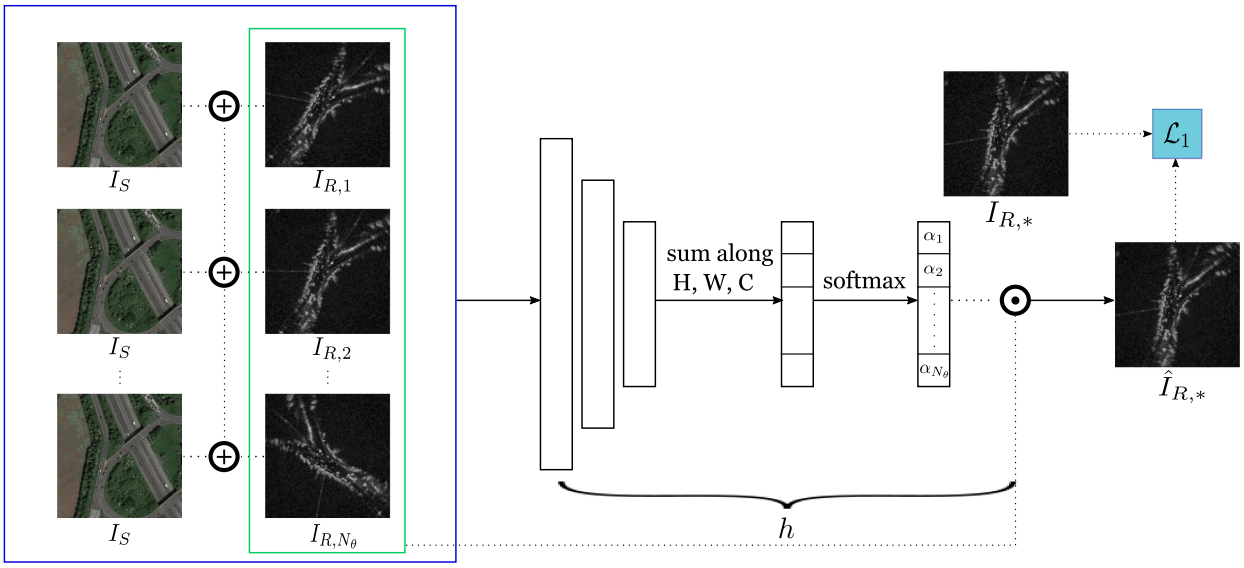


Fig. 2. Given I_S, we rotate the radar image I_R with $\theta \in \{\theta_1, \ldots, \theta_{N_\theta}\}$ to get $\{I_{R,1}, \ldots, I_{R,N_\theta}\}$. Given the concatenated inputs, the network h outputs a linear combination of the rotated images $I_{R,1}, \ldots, I_{R,N_\theta}$, where the coefficients are softmaxed. The output is a rotated radar image with heading matching that of the input satellite image I_S. We use $\oplus$ to denote concatenation and $\odot$ to denote dot product. We can obtain the target $I_{R,*}$ by rotating I_R according to the ground truth heading.

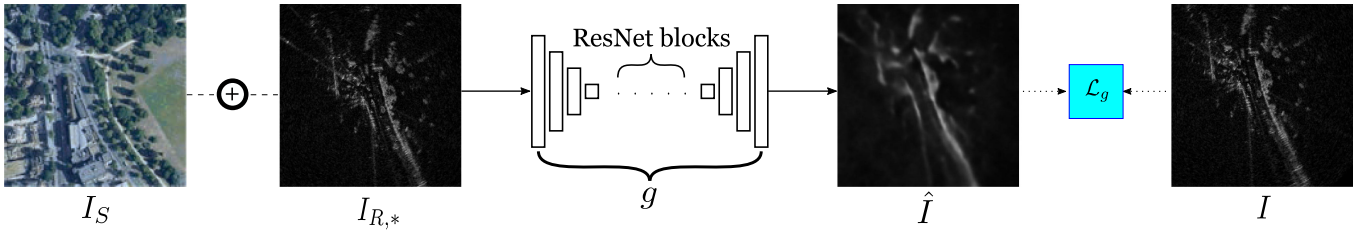


Fig. 3. To pre-train the network g, we rotate I_R to obtain $I_{R,*}$, where the heading of $I_{R,*}$ is the same as I_S. There is still an offset in translation between I_S and $I_{R,*}$. The generator conditions on the pose of I_S and the appearance of $I_{R,*}$, and generates $\hat{I}$ which appears like a radar image, and has the same pose as the input satellite image I_S. We can obtain the target I by shifting $I_{R,*}$ to match the pose as in I_S.

Ideally, the computed coefficient will be close to 1 for the image whose heading is the nearest to the heading in I_S, and close to 0 for the other images in the ensemble. The optimal θ is found by selecting the rotation with the maximum coefficient α_n.

C. Conditional Image Genarator

The generator is conditioned on input pair $(I_{R,*}, I_S)$ and outputs a generated image $\hat{I}$. As depicted in Fig. 3, we use a standard encoder-decoder structure, with 4 down-convolutional layers with stride 2, followed by 9 ResNet blocks at the bottleneck, and 4 up-convolutional layers to recover the same size as the input image.

We pre-train g using an L1 loss on the image as in (7). Empirically, we found this to be sufficient and we do not need to introduce additional losses such as a GAN loss.

$$\mathcal{L}_g = \left\| g(I_{R,*}, I_S) - I \right\|_1 . \tag{7}$$

To pre-train the generator, we feed to the network $I_{R,*}$ rather than $\hat{I}_{R,*}$ outputted by the rotation selector network, since the ground truth $I_{R,*}$ is available at training. As such, the pre-training of the generator g is independent from the pre-training of h as in III-B.

D. Training RSL-Net End-to-End

After pre-training h and g as in Sections III-B and III-C, RSL-Net is trained end-to-end as illustrated in Fig. 4. The generator g now takes $\hat{I}_{R,*} = h(\{I_R\}_{N_\theta}, I_S)$ as input rather than $I_{R,*}$.

The networks e_A and e_B learn to project $\hat{I} = g(\hat{I}_{R,*}, I_S)$ and $\hat{I}_{R,*}$ into a joint embedded domain for pose comparison. We use the same U-Net architecture [39] for both e_A and e_B, with 6 down-convolutional layers and 6 up-convolutional layers and skip connections between the encoder and the decoder stacks. The use of U-Nets is motivated by the observation that much of the measurements in $\hat{I}$ and $\hat{I}_{R,*}$ are useful for pose comparison and should be kept. Regions in the radar image that are distracting for pose comparison, such as returns from reflected signals, or returns on vehicles, are learned to be masked out, as shown in Fig. 1.

The translational offset is found by shifting $e_A(\hat{I})$ by all (x, y) pairs in the solution space, and finding the one that results in the maximum correlation with $e_B(\hat{I}_{R,*})$. As such, we compute (x, y) by finding the global maximum of a discretized, 2D correlation surface over a (x, y) solution space of size $N_x \times N_y$, where N_x and N_y are the number of discretizations in the x and y dimensions.

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

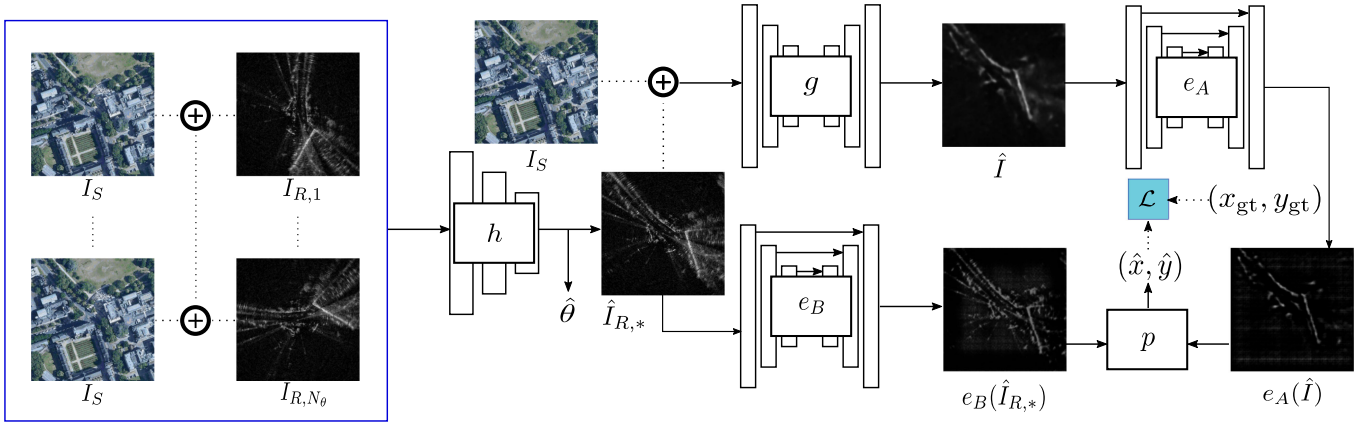


Fig. 4. Having I_S and an ensemble of rotated radar images $\{I_{R,1}, \ldots, I_{R,N_\theta}\}$, we first infer the rotation by passing through the network h, where we produce $\hat{I}_{R,*}$. $\hat{I}_{R,*}$ and I_S are passed through g to generate $\hat{I}$, which has the same pose as I_S. $\hat{I}$ and $\hat{I}_{R,*}$ are projected a joint deep embedding by e_A and e_B, where their translational offset is found. The only loss term in the final stage is a loss on the computed translation.

As noted in [29], computing correlations across all (x, y) translational shifts in the solution space is equivalent to convolving $e_A(\hat{I})$ with $e_B(\hat{I}_{R,*})$. Moreover, this operation can be performed very efficiently in the Fourier domain utilising the convolution theorem [29]:

$$e_A * e_B = \mathcal{F}^{-1}\{\mathcal{F}(e_A) \otimes \mathcal{F}(e_B)\}, \tag{8}$$

where $*$ denotes convolution, $\otimes$ denotes element-wise multiplication, and $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the Fourier and inverse Fourier transforms.

At this point, the optimal solution $\begin{bmatrix}\hat{x} & \hat{y}\end{bmatrix}^T$ is found by applying a softmax over (x, y) across the correlation surface. We denote this process for computing the translational offset in the embeddings as function p, as in (4).

p is differentiable, allowing RSL-Net to be trained end-to-end. For training end-to-end as in Fig. 4, we supervise using only an L1 loss on the computed translation:

$$\mathcal{L} = \left\| p\left(e_A(\hat{I}), e_B(\hat{I}_{R,*})\right) - \begin{bmatrix} x_{\text{gt}} \\ y_{\text{gt}} \end{bmatrix} \right\|_1, \tag{9}$$

where x_{gt} and y_{gt} are the ground truth (x, y) shift.

Since $\hat{I}_{R,*} = h(\{I_R\}_{N_\theta}, I_S)$ and $\hat{I} = g(\hat{I}_{R,*}, I_S)$, (9) optimises the networks h and g as well. We do not include any additional losses such as $\mathcal{L}_h$ (6) or $\mathcal{L}_g$ (7) as in the pre-training stages. This removes the need for hyperparameter tuning on the relative weights among different loss terms.

In [29] the authors search for the optimal (x, y, θ) in the deep embeddings from a 3D solution space, where the 2D embedding is rotated N_θ times with N_θ being the number of discretizations in the θ dimension. In our method, this step is essentially handled previously in the rotation selector network h, where we found the optimal rotation from a discretized solution space $\{\theta_1, \theta_2, \ldots, \theta_{N_\theta}\}$. As such, we only need to search for the optimal translation (x, y) from a 2D solution space in the embedding.

The decision to pre-infer θ stems from the fact that the rotation needs to be uncovered first in order to condition the generator network g, as the input satellite and radar image to g should only be offset in translation, but not in rotation. The work in [29] is not concerned with image generation across sensor modalities, and therefore does not have such limitation. Regardless, we show in Section IV that our computation for θ is accurate despite not performing the search concurrently with (x, y).

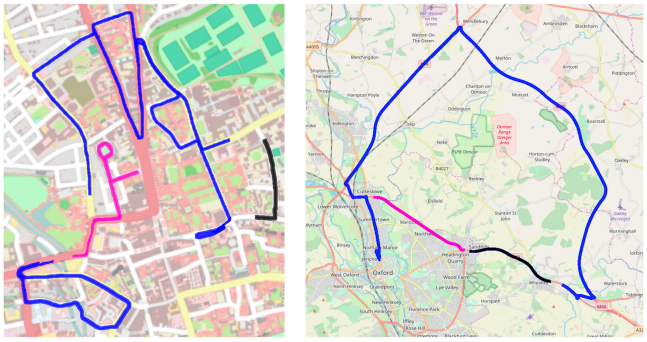


Fig. 5. Routes for the datasets we used are split into training (blue), validation (black), and test (pink), where we discarded data near intersections.

IV. Dataset and Experimental Setup

Our pipeline is validated using the publicly available Oxford Radar RobotCar Dataset [40], [41], which features an urban setting, and our own dataset that consists of highway driving in rural regions.

We used three sequences from the RobotCar Dataset for training, where we split the route into training, validation, and test sets, as shown in Fig. 5. For simplicity, only the test data from the first sequence was used for evaluation, as all sequences take place on the same route. Similarly, we also split the Highway Dataset into training, validation, and test sets, shown in Fig. 5. We discarded data near the intersections between training, validation, and test data such that the radar does not observe any of the test set scenes in the training and validation data. We test on every 5th frame of the test set routes, where the radar frames arrive at $4\,\text{Hz}$. This results in 201 test images for the RobotCar Dataset and 261 test samples for the Highway Dataset, spanning a total distance of near $10\,\text{km}$.

The RobotCar Dataset and our Highway Dataset are collected with different models of the radar, and exhibit slightly different sensing and noise characteristics. Despite this, we combine

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

TABLE I
MEAN ERROR OF RSL-NET AND OTHER METHODS ON THE ROBOTCAR DATASET AND HIGHWAY DATASET. e_x, e_y, AND e_θ DENOTES THE MEAN ERROR IN x, y, AND θ, RESPECTIVELY. RS IS ABBREVIATION FOR ROTATION SELECTOR; CV IS ABBREVIATION FOR COST VOLUME. THE MEAN ERROR IS COMPUTED AS $e = \frac{\sum \|d_i\|}{N}$, WHERE d_i IS THE DIFFERENCE OF THE i^{TH} SAMPLE FROM GROUND TRUTH. WE USED 201 IMAGES IN THE TEST SET FOR ROBOTCAR AND 261 FOR HIGHWAY

	e_θ (°)	e_x (m)	e_y (m)
RobotCar (ours)	**3.12**	**2.74**	**4.26**
Highway (ours)	**2.66**	**8.13**	**5.40**
RobotCar (Pix2Pix)	6.15	4.07	6.25
Highway (Pix2Pix)	5.98	8.70	5.82
RobotCar (H-Nets)	4.42	11.65	8.74
Highway (H-Nets)	6.64	11.77	9.14
RobotCar (RSL-Net with $g(I_S) \rightarrow I$)	3.12	10.25	10.67
Highway (RSL-Net with $g(I_S) \rightarrow I$)	2.66	10.33	6.84
RobotCar (RS + regression)	3.12	4.10	5.51
Highway (RS + regression)	2.66	9.28	6.67
RobotCar (RS + STN)	3.12	4.35	4.87
Highway (RS + STN)	2.66	8.66	5.87
RobotCar (direct regression)	23.56	9.70	10.87
Highway (direct regression)	24.26	10.85	9.43
RobotCar (Canny edges + CV)	12.53	17.75	35.53
Highway (Canny edges + CV)	10.60	36.73	42.67

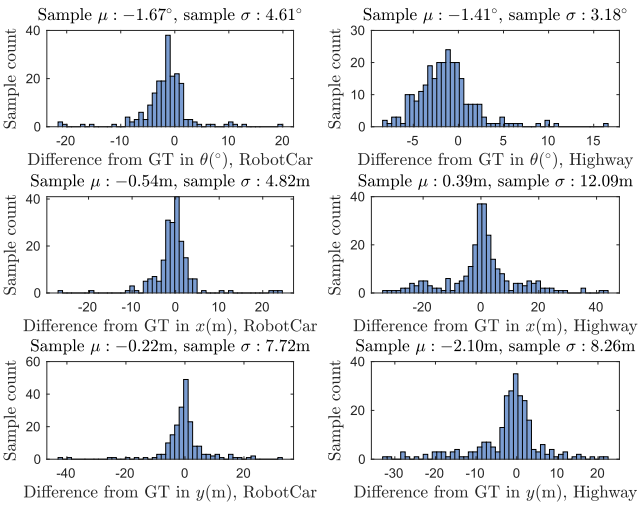


Fig. 6. Histogram distribution for errors in x, y, and θ for the RobotCar and Highway datasets, along with sample mean μ and standard deviation σ.

the two datasets together for training, and demonstrate high accuracy on both test sets. The combined training set consists of 23440 images which spans 98 minutes of driving. Ground truth latitude, longitude, and heading are available for both datasets with an on-board Novatel GNSS.

For all of our experiments, we use radar and satellite images with size 256×256, with a shared resolution of 0.8665 m/pixel.

To simulate a coarse initial position estimate good enough for place recognition but not for metric localisation, we add artificial, uniformly random offsets to the ground truth position when querying for the satellite image, such that the satellite image is off by up to 25 pixels (21.66 m) in either x and y directions. However, we do not limit the solution space for x and y to be 25 pixels, but rather use a search space of $[-128, 128]$ pixels for both x and y. The satellite images are queried using

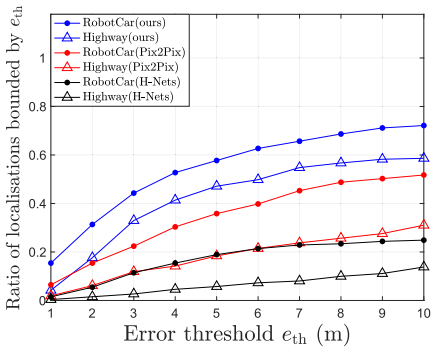


Fig. 7. Ratio of localisations where both e_x and e_y are less than e_{th}, and e_θ is less than 5°.

Google Maps Static API.[2] Similarly, we assume a coarse initial heading estimate is available, such that the solution space for θ does not need to span $[-\pi, \pi]$. This not only reduces the computational effort needed to search for θ, but is also crucial against heading ambiguity due to scene symmetry, for example down a straight road where the radar image looks almost invariant after a rotation by π. In our experiments, we add an artificial, uniformly random offset in the range $\theta_0 \in [\frac{-\pi}{8}, \frac{\pi}{8}]$ to the ground truth heading to simulate a coarse initial estimate, so that the rotation selector network h searches for θ in the solution space of $[\theta_0 - \frac{\pi}{8}, \theta_0 + \frac{\pi}{8}]$, at intervals of 1°.

V. EXPERIMENTAL RESULTS

The localisation results on the test data of the RobotCar and Highway datasets are summarised in Table I and Fig. 6. Our method achieves a mean error of a few degrees in θ. We have an error of a few metres in (x, y) on the RobotCar Dataset, but the translation error is larger on the Highway Dataset. This is due to highway settings offering less geometric constraints for pose estimation such as buildings and structures than a urban environment. Given a specific threshold e_{th} on translation, we compute the ratio of localisations where both e_x and e_y are less than e_{th}, and e_θ is less than 5°, shown in Fig. 7. The average computation time is 99.1 ms per localisation on a 1080 Ti GPU, allowing us to operate in real-time.

VI. COMPARISON AGAINST OTHER METHODS

We implemented pipelines using Pix2Pix [30] and harmonic networks (H-Nets) [35], [42]. Without a rotation selector to pre-infer the heading, the generator was pre-trained to condition on the radar image rotated with a coarse initial heading estimate θ_0, $g : (I_{R,0}, I_S) \rightarrow I$, using a conditional GAN. We also learn the deep embeddings e_A and e_B for pose computation, but rather search for the optimal (x, y, θ) from a 3D correlation volume. The results are shown in Table I and Fig. 7. The use of harmonic networks resulted in small e_θ on the RobotCar Dataset, possibly because of the rotational equivariance property, but rather large e_x and e_y. For Pix2Pix, we also show examples of the synthetic radar image when the generator does not condition on radar data, $g : (I_S) \rightarrow I$, which leads to inferior results than learning $g : (I_{R,0}, I_S) \rightarrow I$, shown in Fig. 8.

[2]Google Maps Static (API) (2019): https://developers.google.com/maps/documentation/maps-static/

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

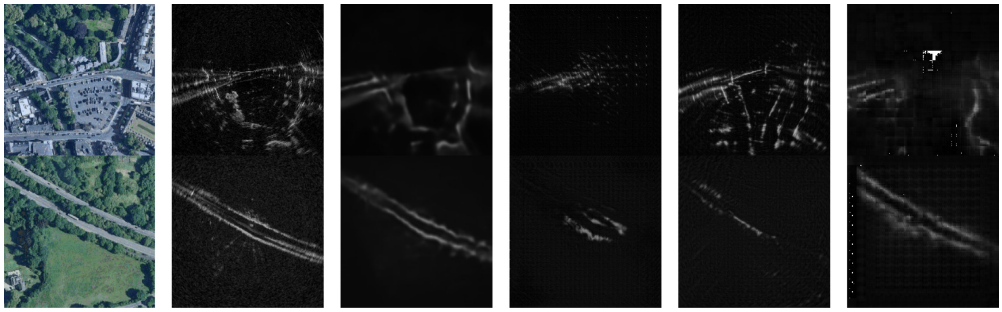


Fig. 8. (a) I_S. (b) Ground truth I. (c) $\hat{I}$ generated using RSL-Net. (d) and (e) $\hat{I}$ using Pix2Pix with $g(I_S)$ and $g(I_{R,0}, I_S)$. (f) $\hat{I}$ using H-Nets with $g(I_{R,0}, I_S)$. For Pix2Pix, we use the same generator architecture as in RSL-Net. For H-Nets, we use an architecture with a similar number of parameters.

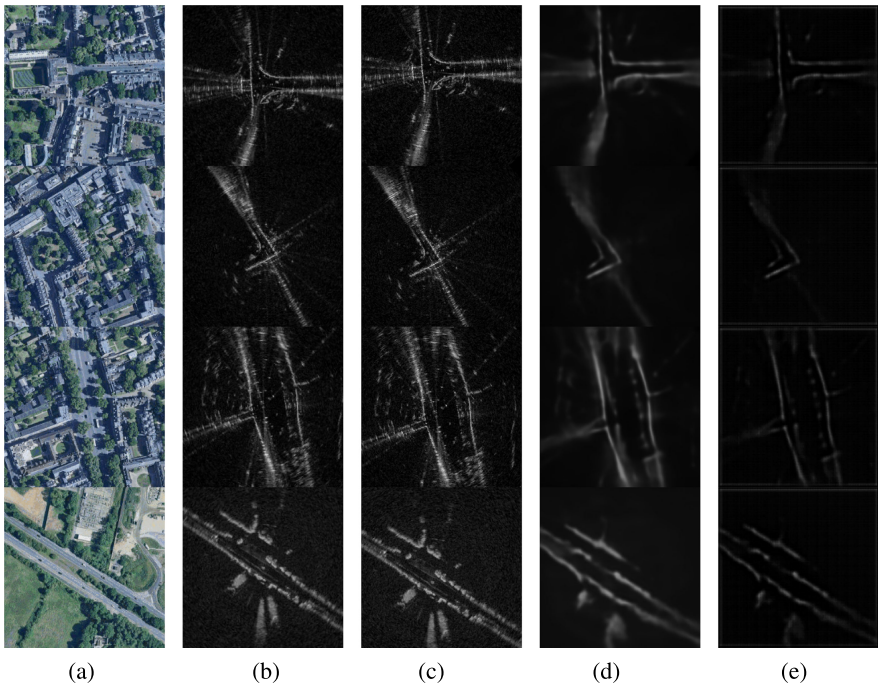


Fig. 9. Additional qualitative results from the test set of the RobotCar Dataset (rows 1-3) and the Highway Dataset (row 4). (a) Satellite image I_S. (b) Radar image with rotation offset uncovered, $\hat{I}_{R,*}$. (c) Ground truth I. (d) Synthetic radar image $\hat{I}$.(e) Deep embedding $e_A(\hat{I})$.

As shown in Fig. 8, without conditioning using radar data, the synthetic images in (d) can appear different than what the radar actually measures, making pose estimation difficult. While they are more realistic when conditioning on $I_{R,0}$, the results in (e) and (f) are still inferior comparing to (c), as the generator in RSL-Net utilises $\hat{I}_{R,*}$ which has approximately the same heading as the input satellite image.

An experiment was performed where we modified the generator in RSL-Net to learn $g(I_S) \rightarrow I$ instead, and kept other modules the same, to further demonstrate the added benefit of conditioning the generator using live radar data.

To demonstrate the advantage of our correlation-based search for finding the translation offset over regression, we trained a network that takes in $\hat{I}_{R,*}$ and $\hat{I}$ as inputs, and directly regresses the (x, y) offset, where the last layer is a fully connected layer. We have also benchmarked against a baseline Spatial Transformer Net (STN) [43] method, where given $\hat{I}_{R,*}$ and $\hat{I}$, the network regresses the (x, y) offset, and transforms $\hat{I}_{R,*}$ to register onto $\hat{I}$. While both methods regress (x, y), the first regression method is supervised by a loss on pose and the STN method is supervised with an L1 loss on the registered images. The results are shown in Table I. To ensure a fair comparison, the rotation is pre-inferred using the rotation selector as in RSL-Net in order to generate $\hat{I}$ correctly.

We have also experimented with other simpler methods, such as directly regressing (x, y, θ) given the input satellite-radar pair, or finding the pose using Canny edges extracted. These methods led to poor results, as shown in Table I. We believe this is due to radar and satellite images being drastically different forms of data, and requires careful treatment prior to pose computation.

Additional examples of images at various stages of RSL-Net are shown in Fig. 9.

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

VII. Conclusion

In this work, we introduce a novel framework to learn the metric localisation of a ground range sensor using overhead imagery, namely RSL-Net. From a coarse initial pose estimate, RSL-Net computes a refined metric localisation by comparing live radar images against Google satellite images. RSL-Net allows a vehicle to localise in real-time using an on-board radar without relying on any prior radar data, offering a fallback solution when the vehicle fails to localise against prior sensor maps. The experimental validation demonstrates the capability for RSL-Net to perform even outside of well-structured, urban environments.

References

[1] W. Churchill and P. Newman, "Experience-based navigation for long-term localisation," *Int. J. Robot. Res.*, vol. 32, no. 14, pp. 1645–1661, 2013.

[2] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, and R. Siegwart, "Long-term 3d map maintenance in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 3712–3719.

[3] H. Porav, W. Maddern, and P. Newman, "Adversarial training for adverse conditions: Robust metric localisation using appearance transfer," in *Proc. IEEE Int. Conf. Robot. Autom.*, Brisbane, QLD, 2018, pp. 1011–1018.

[4] S. H. Cen and P. Newman, "Precise ego-motion estimation with millimeter-wave radar under diverse and challenging conditions," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 1–8.

[5] M. Noda *et al.*, "Vehicle ego-localization by matching in-vehicle camera images to an aerial image," in *Proc. Asian Conf. Comput. Vision*, 2010, pp. 163–173.

[6] K. Y. K. Leung, C. M. Clark, and J. P. Huissoon, "Localization in Urban environments by matching ground level video images with an aerial image," in *Proc. IEEE Int. Conf. Robot. Autom.*, Pasadena, CA, USA, 2008, pp. 551–556.

[7] O. Pink, "Visual map matching and localization using a global feature map," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. Workshops*. Anchorage, AK, USA, 2008, pp. 1–7.

[8] A. Viswanathan, B. R. Pires, and D. Huber, "Vision based robot localization by ground to satellite matching in gps-denied situations," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Chicago, IL, USA, 2014, pp. 192–198.

[9] T. Senlet and A. Elgammal, "A framework for global vehicle localization using stereo images and satellite and road maps," in *Proc. IEEE Int. Conf. Comput. Vision Workshops*. Barcelona, 2011, pp. 2034–2041.

[10] R. Kümmerle, B. Steder, C. Dornhege, A. Kleiner, G. Grisetti, and W. Burgard, "Large scale graph-based slam using aerial images as prior information," *Auton. Robots*, vol. 30, no. 1, pp. 25–39, 2011.

[11] L. De Paula Veronese *et al.*, "Re-Emission and satellite aerial maps applied to vehicle localization on urban environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Hamburg, 2015, pp. 4285–4290.

[12] X. Wang, S. Vozar, and E. Olson, "Flag: Feature-Based localization between air and ground," in *Proc. IEEE Int. Conf. Robot. Autom.*, Singapore, 2017, pp. 3178–3184.

[13] C. Forster, M. Pizzoli, and D. Scaramuzza, "Air-Ground localization and map augmentation using monocular dense reconstruction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, 2013, pp. 3971–3978.

[14] H. Chu, H. Mei, M. Bansal, and M. R. Walter, "Accurate vision-based vehicle localization using satellite imagery," in *Proc. Neural Informat. Process. Syst., workshop Transfer Multi-Task Learn.*, 2015.

[15] D.-K. Kim and M. R. Walter, "Satellite image-based localization via learned embeddings," in *Proc. IEEE Int. Conf. Robot. Autom.*, Singapore, 2017, pp. 2073–2080.

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

[17] S. Workman, R. Souvenir, and N. Jacobs, "Wide-Area image geolocalization with aerial reference imagery," in *Proc. IEEE Int. Conf. Comput. Vision*, Santiago, 2015, pp. 3961–3969.

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

[19] M. Hentschel and B. Wagner, "Autonomous robot navigation based on openstreetmap geodata," in *13th Int. IEEE Conf. Intell. Transp. Syst.*, Funchal, 2010, pp. 1645–1650.

[20] M. A. Brubaker, A. Geiger, and R. Urtasun, "Lost! leveraging the crowd for probabilistic visual self-localization," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Portland, OR, 2013, pp. 3057–3064.

[21] G. Floros, B. Van Der Zander, and B. Leibe, "Openstreetslam: Global vehicle localization using openstreetmaps," in *Proc. IEEE Int. Conf. Robot. Autom.*, Karlsruhe, 2013, pp. 1054–1059.

[22] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *IEEE Int. Conf. Comput. Vision*, Santiago, 2015, pp. 2938–2946.

[23] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Honolulu, HI, 2017, pp. 6555–6564.

[24] Z. Laskar, I. Melekhov, S. Kalia, and J. Kannala, "Camera relocalization by computing pairwise relative poses using convolutional neural network," in *Proc. IEEE Int. Conf. Comput. Vision Workshops*, Venice, 2017, pp. 920–929.

[25] E. Brachmann and C. Rother, "Learning less is more-6d camera localization via 3d surface regression," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, Salt Lake City, UT, 2018, pp. 4654–4662.

[26] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz, "Geometry-Aware learning of maps for camera localization," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, Salt Lake City, UT, 2018, pp. 2616–2625.

[27] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, Long Beach, CA, USA, 2019, pp. 12708–12717.

[28] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-net: Towards learning based lidar localization for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, Long Beach, CA, USA, 2019, pp. 6382–6391.

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

[30] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Honolulu, HI, 2017, pp. 5967–5976.

[31] J.-Y. Zhu *et al.*, "Toward multimodal image-to-image translation," in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 465–476.

[32] G. Perarnau, J. Van De Weijer, B. Raducanu, and J. M. Álvarez, "Invertible conditional gans for image editing," in *Proc. Neural Informat. Process. Syst., Workshop Adversarial Training*, 2016.

[33] J. Lin, Y. Xia, T. Qin, Z. Chen, and T.-Y. Liu, "Conditional image-to-image translation," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, Salt Lake City, UT, 2018, pp. 5524–5532.

[34] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool, "Pose guided person image generation," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, 2017, pp. 406–416.

[35] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, "Harmonic networks: Deep translation and rotation equivariance," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Honolulu, HI, 2017, pp. 7168–7177.

[36] M. Weiler, F. A. Hamprecht, and M. Storath, "Learning steerable filters for rotation equivariant CNNs," in *IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, Salt Lake City, 2018, pp. 849–858.

[37] C. Esteves, C. Allen-Blanchette, X. Zhou, and K. Daniilidis, "Polar transformer networks," in *Proc. 6th Int. Conf. Learn. Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=HktRlUlAZ

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Las Vegas, NV, 2016, pp. 770–778.

[39] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Med. Image Comput. Comput.-Assisted Intervention*, 2015, pp. 234–241.

[40] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset," 2019. [Online]. Available: https://arxiv.org/pdf/1909.01300

[41] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000 km: The Oxford RobotCar dataset," *Int. J. Robot. Res.* vol. 36, no. 1, pp. 3–15, 2017. [Online]. Available: https://dx.doi.org/10.1177/0278364916679498

[42] M. Tyszkiewicz, "Reimplementation of harmonic networks in pytorch," 2019. [Online]. Available: https://github.com/jatentaki/harmonic

[43] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Proc. Conf. Workshop Neural Inf. Process. Syst.*, 2015, pp. 2017–2025.

Authorized licensed use limited to: Bodleian Libraries of the University of Oxford. Downloaded on December 27,2022 at 03:51:19 UTC from IEEE Xplore. Restrictions apply.

4

Self-Supervised Learning for Using Overhead Imagery as Maps in Outdoor Range Sensor Localization

This work was published as a journal article in The International Journal of Robotics Research in December 2021. A conference version of this article, "Self-Supervised Localisation between Range Sensors and Overhead Imagery" was published and presented at Robotics: Science and System XVI in July 2020, and included in appendix B for completeness.

Background and Motivation

The work in Chapter 3 learns the metric localisation between radar and overhead imagery by generating synthetic radar images. This method requires pixel-wise aligned pairs of radar and satellite images for training, which in turn requires accurate ground truth with sub-metre level precision. Obtaining accurate ground truth requires a high-end on-board GPS/INS, and possibly post-processing in a batch optimisation with SLAM solutions, which can be costly and time consuming.

To this end, in this chapter we present a self-supervised approach to learning the metric localisation between range sensor and overhead imagery that no longer requires metrically accurate ground truth.

Self-Supervised Learning for Using Overhead Imagery as Maps in Outdoor Range Sensor Localisation

Tim Y. Tang[1], Daniele De Martini[1], Shangzhe Wu[2], and Paul Newman[1]

Abstract

Traditional approaches to outdoor vehicle localisation assume a reliable, prior map is available, typically built using the same sensor suite as the on-board sensors used during localisation. This work makes a different assumption. It assumes that an overhead image of the workspace is available and utilises that as a map for use for range-based sensor localisation by a vehicle. Here, range-based sensors are radars and lidars. Our motivation is simple, off-the-shelf, publicly available overhead imagery such as Google satellite images can be a ubiquitous, cheap, and powerful tool for vehicle localisation when a usable prior sensor map is unavailable, inconvenient, or expensive. The challenge to be addressed is that overhead images are clearly not directly comparable to data from ground range sensors because of their starkly different modalities. We present a learned metric localisation method that not only handles the modality difference, but is also cheap to train, learning in a self-supervised fashion without requiring metrically accurate ground truth. By evaluating across multiple real-world datasets, we demonstrate the robustness and versatility of our method for various sensor configurations in cross-modality localisation, achieving localisation errors on-par with a prior supervised approach while requiring no pixel-wise aligned ground truth for supervision at training. We pay particular attention to the use of millimetre-wave radar, which, owing to its complex interaction with the scene and its immunity to weather and lighting conditions, makes for a compelling and valuable use case.

Keywords

localisation, cross-modality localisation, deep learning, self-supervised learning

1 Introduction

The ability to localise relative to an operating environment is central to robot autonomy. Localisation using range sensors, such as lidars (Levinson and Thrun 2010; Wolcott and Eustice 2015) and, more recently, scanning millimetre-wave radars (Park et al. 2019; Saftescu et al. 2020; De Martini et al. 2020), is an established proposition. Both are immune to changing lighting conditions and directly measure scale, while the latter adds resilience to weather conditions.

Current approaches to robot localisation typically rely on a prior map built using a sensor configuration that will also be equipped on-board, for example a laser map for laser-based localisation. This paper looks at an alternative method. Public overhead imagery such as satellite images can be a reliable map source, as they are readily available, and often capture information also observable, albeit perhaps in some complex or incomplete way, by sensors on the ground. We can pose the localisation problem in a natural way: finding the pixel location of a sensor in an overhead (satellite) image given range data taken from the ground. The task is, however, non-trivial because of the drastic modality difference between satellite images and sparse, ground-based radar or lidar.

Recent work on learning to localise a ground scanning radar against satellite images by Tang et al. (2020b) provides a promising direction which addresses the modality difference by first generating from a satellite image a synthetic radar image. The synthetic image can then be "compared" against live radar data, expressed as 2D images from a "birds-eye" perspective, for pose estimation. Such an approach learns metric, cross-modality localisation in an end-to-end fashion, and therefore does not require hand-crafted features limited to a specific environment.

The method in Tang et al. (2020b) trains a multi-stage network, and needs pixel-wise aligned radar and satellite image pairs for supervision at all stages. This in turn relies on sub-metre and sub-degree accurate ground truth position and heading signals, which in practice requires high-end GPS/Inertial Navigation System (INS) and possibly bundle adjustment along with other on-board sensor solutions, bringing in burdens in terms of cost and time consumption.

To address this issue, building on Tang et al. (2020b), we propose a method for localising against satellite imagery that is learned in a self-supervised fashion. The core idea is still to generate a synthetic image with the appearance and observed scenes of a live range sensor image, but pixel-wise aligned with the satellite image. Yet, we relax the requirement on pixel-wise aligned data pairs and assume only a coarse initial pose estimate is available from a place recognition system, such that there is reasonable overlap between the live

[1]Mobile Robotics Group, University of Oxford
[2]Visual Geometry Group, University of Oxford

Corresponding author:
Tim Y. Tang, Mobile Robotics Group, Oxford Robotics Institute, University of Oxford, Oxford, OX2 6NN, United Kingdom
Email: ttang@robots.ox.ac.uk

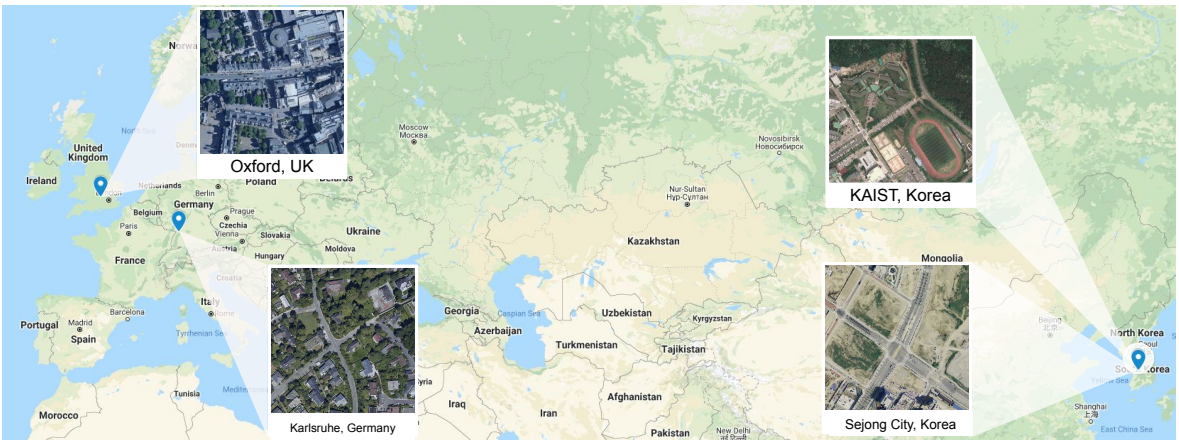


Figure 1. Our method is demonstrated on datasets collected around different locations, at various types of settings including urban (Oxford, UK), residential (Karlsruhe, Germany), campus (KAIST, Korea), and highway (Sejong City, Korea).

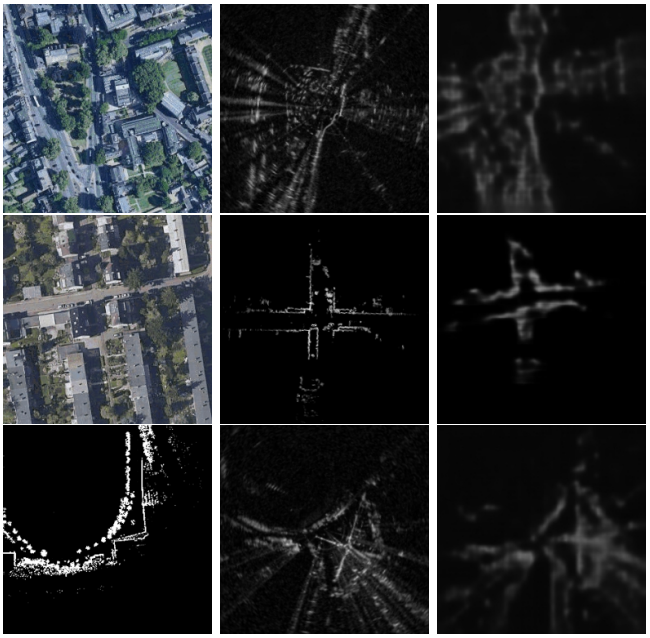


Figure 2. Given a map image of modality $\mathcal{A}$ (left) and a live data image of modality $\mathcal{B}$ (middle), we wish to find the unknown $SE(2)$ offset between them. To do so, our method generates a synthetic image of modality $\mathcal{B}$ (right) that is pixel-wise aligned with the map image, but contains the same appearance and observed scenes as the live data image. Top: localising radar data against satellite imagery. Middle: localising lidar data against satellite imagery. Bottom: localising radar data against prior lidar map.

ground sensor field-of-view and a queried satellite image. Our method does not solve the global localisation problem. Instead, given a coarse initial pose estimate from place recognition, our method solves the metric localisation of a range sensor using overhead imagery, providing a refined metric pose.

Vitally, here we make no use of metrically accurate ground truth for training. Note also that although designed for localising against satellite imagery, our method can naturally handle other forms of cross-modality registration, such as localising a radar against a prior lidar map. Figure 2 shows synthetic images generated by our method used for pose estimation.

To the best of our knowledge, our proposed method is the first to learn the cross-modality, metric localisation of a range sensor in a self-supervised fashion. Our method is validated experimentally on multiple datasets and achieves performances on-par with a state-of-the-art, supervised approach.

This paper is an extended version of our prior work (Tang et al. 2020a). The improvements include a more detailed explanation for the motivation of our method to tackle the problem of localising range sensors using aerial imagery (Section 3) and a more thorough description of our method (Section 4). For experimental validation (Section 6), we present additional qualitative results, an ablation study with reduced training data, a study on the trade-offs between network width and depth and solution quality, and an analysis on the choice of image resolution. We also introduce an introspective strategy at inference time to handle initial pose offsets larger than in the training data. Finally, we show with unsupervised domain adaptation, models trained using radar data can be utilised for localisation between lidar and overhead imagery, and vice versa.

2 Related Work

Our approach is related not only to other works in the field of localisation using overhead imagery and the general theme of cross-modality localisation, but also to learned methods for range sensor state estimation and unsupervised image generation. We provide broad coverage of the most relevant research in these subjects in this section.

2.1 Localisation Using Overhead Images

Localisation using aerial or overhead images has been of interest for the community for over a decade. The methods in Leung et al. (2008); Li et al. (2014); Parsley and Julier (2010) localise a ground camera using aerial images, by detecting Canny edges from aerial imagery, and matching against lines detected by a ground camera. Several other vision-based approaches project the ground camera images to a top-down perspective via a homography, and compare against the aerial imagery by detecting lane markings (Pink 2008), SURF features (Noda et al. 2010), or dense matching (Senlet and Elgammal 2011). Recent work by Chebrolu et al. (2019) localises a ground robot in a crop field by matching camera features against landmarks from an aerial map, and explicitly incorporates semantics of crops to reduce ambiguity.

Metric localisation of range sensors or point-clouds against overhead imagery requires further pre-processing due to the modality difference. Kaminsky et al. (2009) projected point-clouds into images and matched against binary edge images from overhead imagery. The method in Kaminsky et al. (2009) also constructs a ray image by ray-tracing each point, and introduces a free-space cost to aid the image registration. The work by de Paula Veronese et al. (2015) accumulates several lidar scans to produce dense lidar intensity images, which are then matched against satellite images utilising Normalised Mutual Information. Similar as Kaminsky et al. (2009), several other methods also pre-process the aerial image before matching against ground laser observations, for example using edge detection (Kümmerle et al. 2011) or semantic segmentation (Dogruer et al. 2010). In contrast to these approaches, our method directly learns the metric localisation of a range sensor end-to-end, without the need for careful pre-processing or manual feature definition.

Closely related to our method is the seminal work on learning to localise a ground radar against satellite imagery by Tang et al. (2020b). As discussed previously, the method in Tang et al. (2020b) requires pixel-wise aligned ground truth for supervision, while our method is self-supervised.

2.2 Cross-Modality Localisation

Other forms of cross-modality localisation are also heavily studied by the community. Several works propose to localise a forward-facing camera against a prior 3D point-cloud map (Wolcott and Eustice 2014; Caselitz et al. 2016; Xu et al. 2017). Carle and Barfoot (2010) localised a ground laser scanner against an orbital elevation map. The works in Wang et al. (2019); Boniardi et al. (2017); Wang et al. (2017); Mielle et al. (2019) localise an indoor lidar or stereo camera against architectural floor plans. Brubaker et al. (2013) and Floros et al. (2013) concurrently proposed matching visual odometry paths to road layouts from OpenStreetMap for localisation.

2.3 Learning-based State Estimation for Range Sensors

A number of recent works were proposed for learning the odometry or localisation of lidars. Barsan et al. (2018) represented lidar data as intensity images, and learned a deep embedding specifically for metric localisation that can be used for direct comparison of live lidar data against a previously-built lidar map. Other methods such as Cho et al. (2019); Li et al. (2019), instead, learn deep lidar odometry by projecting lidar point-clouds into other representations before passing through the network. Lu et al. (2019b) learned descriptors from input point-clouds, and utilised 3D Convolutional Neural Networks (CNNs) for solving $SE(2)$ metric localisation by searching in a 3D cost volume. In their later work, Lu et al. (2019a) proposed a method to learn $SE(3)$ lidar point-cloud registration end-to-end.

As an emerging sensor for outdoor state estimation, learning-based methods were proposed for scanning Frequency-Modulated Continuous-Wave (FMCW) radars. Aldera et al. (2019) utilised an encoder-decoder on polar image representation of radar scans to reject superfluous points for decreasing computation time in the classical radar odometry method described by Cen and Newman (2019). Barnes et al. (2019) learned image-based radar odometry by masking out regions distracting for pose estimation. Barnes and Posner (2020) learned point-based radar odometry by detecting key-points from radar images. Saftescu et al. (2020) encoded images of polar radar scans through a rotation-invariant architecture to perform topological localisation (place recognition), which can then be used for querying a previously-built map (De Martini et al. 2020). These methods, however, are designed to compare data of the same sensor type, and do not address modality difference. Our approach is similar to Barsan et al. (2018); Cho et al. (2019); Aldera et al. (2019); Barnes et al. (2019); Weston et al. (2019); Saftescu et al. (2020); Barnes and Posner (2020); Broome et al. (2020) in that we also represent range sensor data as 2D images.

2.4 Unsupervised Image Generation

A fundamental step in our approach is the generation of a synthetic image before pose computation, where there is no pixel-wise aligned target image for supervision. CycleGAN (Zhu et al. 2017) achieves unsupervised image-to-image transfer between two domains $\mathcal{X}$ and $\mathcal{Y}$, by learning two pairs of generators and discriminators, and enforcing cycle-consistency when an image is mapped from $\mathcal{X}$ to $\mathcal{Y}$ and back from $\mathcal{Y}$ to $\mathcal{X}$, and vice versa. Other methods (Liu et al. 2017; Lee et al. 2018) also utilise cycle-consistency but make different assumptions on how the latent spaces of the two domains are treated. While these methods are concerned with generating photo-realistic images, we are interested in the problem of metric localisation. As such, we need to explicitly encourage the synthetic image to contain information appropriate for pose estimation, rather than aiming for photo-realistic reconstruction.

Several prior works are also geometry-aware. The methods by Shu et al. (2018); Wu et al. (2019); Xing et al. (2019) use separate encoders and/or decoders to disentangle geometry and appearance. The results are networks that can separately interpolate the geometry and appearance of the output images. Similarly, our method separately encodes information about the appearance and the relative pose offset, resulting in an architecture where the two are explicitly disentangled.

3 Overview and Motivation

We seek to solve for the $SE(2)$ pose between a map image of modality $\mathcal{A}$ and a live data image of modality $\mathcal{B}$. Our main focus is when modality $\mathcal{A}$ is satellite imagery, while modality $\mathcal{B}$ are range sensor data represented as an image.

Previously, Radar-Satellite Localisation Network (RSL-Net) (Tang et al. 2020b) was proposed to solve for the metric localisation between matched pairs of radar and satellite images. In particular, it aims to generate a synthetic image that preserves the appearance and observed scenes of the live radar image, and is pixel-wise aligned with the paired satellite image. The synthetic image and the live radar image are then projected onto deep embeddings, where their pose offset is found by maximising a correlation surface. We

follow the same general approach, but, unlike RSL-Net, our method learns in a self-supervised fashion.

3.1 Hand-crafting Features vs. Learning

Some of the works listed in Sections 2.1 and 2.2 can achieve decent accuracy on localising a ground range sensor against aerial imagery. However, they typically rely on pre-processing the aerial images using hand-crafted features or transformations designed for a specific set-up and may not generalise to other sensors or different environments. For example, Kümmerle et al. (2011) focus on detecting edges from a campus dominated by buildings. de Paula Veronese et al. (2015) directly match accumulated lidar intensity images against aerial imagery without pre-processing, yet this is inappropriate for radars due to the complexity of their return signals.

Our data-driven approach instead learns to directly infer the geometric relationship across modalities, remaining free of hand-crafted features. We show in Section 6 the robustness of our method when localising against satellite imagery in various types of scenes, including urban, residential, campus, and highway (Figure 1).

3.2 Generating Images vs. Direct Regression

A naïve approach would be to take a satellite image and a live data image as inputs, and directly regress the pose. Yet, as shown in Tang et al. (2020b), this led to poor results even for the supervised case. Our hypothesis is that when the two images are starkly different in appearance and observed scenes, the problem becomes too complex for direct regression to succeed given current techniques.

Generating synthetic images prior to pose estimation brings two advantages over directly regressing the pose. First, it is a simpler and less ill-posed problem than directly regressing the pose, particularly because we can utilise the live data image to condition the generation. Secondly, the image generation loss is distributed over an entire image of $H \times W$ pixels, where H and W are height and width, instead of on just three pose parameters that describe an $SE(2)$ displacement (x, y, and θ), introducing greater constraints during optimisation.

3.3 Conditional Image Generation

We tackle the synthesis of an image of the live data modality $\mathcal{B}$ from one of the map modality $\mathcal{A}$ as a conditional image generation task – i.e. taking both a map (e.g., satellite) image and a live data image as inputs. An alternative approach is to learn a domain adaptation directly from $\mathcal{A}$ to $\mathcal{B}$, without conditioning on the live data image – e.g., standard image-to-image transfer such as CycleGAN (Zhu et al. 2017).

In practice, the map (e.g., satellite) image is a denser representation of the environment than a frame of data captured by a range sensor. Only a fraction of the scenes captured in a satellite map is present in a ground sensor field-of-view, resulting in the scan to appear drastically different depending on the sensor pose. In other words, the mapping from a satellite image to a range sensor image is not one-to-one, but one-to-many, as illustrated in Figure 3. Figure 4 demonstrates this concept on real data: the overlapping regions of the two satellite images are identical, while the two radar images observe different portions of the scene and as such appear drastically different.

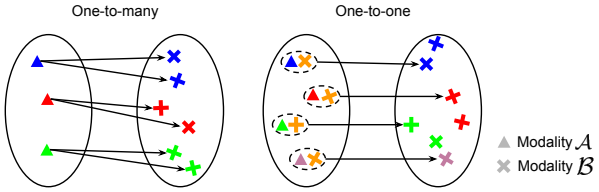


Figure 3. A one-to-many mapping (left) vs. a one-to-one mapping (right). Left: the mapping from modality $\mathcal{A}$ to modality $\mathcal{B}$ preserves color, but is ambiguous in orientation of the output, resulting in a one-to-many mapping, and is therefore not a function. Right: augmenting the input with an element of $\mathcal{B}$ offers additional constraint in orientation, resulting in a one-to-one mapping as the mapping is now unambiguous in both color and orientation. Note that the mapping on the right is one-to-one, but not necessarily *surjective*.

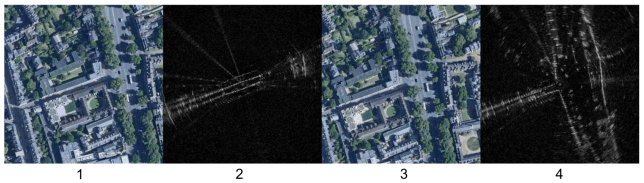


Figure 4. Two radar images captured 15 seconds apart from each other (2 & 4), pixel-wise aligned with satellite images (1 & 3). Though the overlapping scenes in the satellite images are identical, the radar scans appear significantly different, as they capture different regions in their field-of-view.

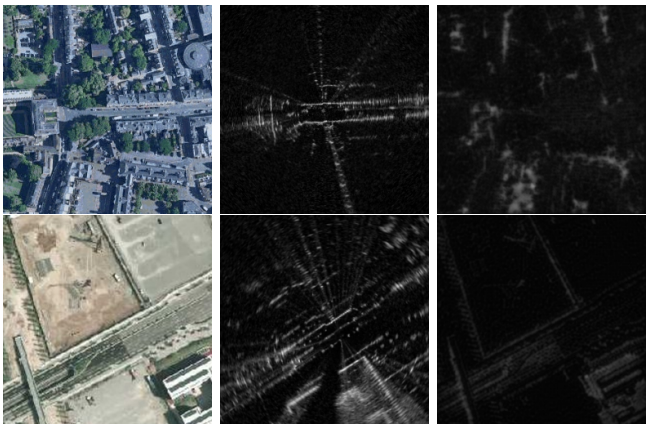


Figure 5. Results of CycleGAN: satellite image (left), live radar image pixel-wise aligned with the satellite image (middle), synthetic radar image (right). There is no explicit constraint on which regions of the input satellite image will appear in the output synthetic image. As a result, this leads to large localisation error as the synthetic image does not contain scenes observed by the live radar image.

By using a naïve image-to-image transfer approach, there is no guarantee for the generated image to contain regions of the scene that are useful for pose comparison against the live data image. Figure 5 shows examples of images generated using CycleGAN (Zhu et al. 2017), where the synthetic image highlights different scenes than what are observed by the live data image. The issue with observability or occlusion can potentially be handled by ray-tracing, such as in Kaminsky et al. (2009). However, not only is this computationally expensive, it does not apply to FMCW radars which have multiple range returns per azimuth (see Barnes et al. (2020) and Cen and Newman (2019) for more details on the sensing characteristics of FMCW radars).

Our approach inherently addresses this problem: by conditioning the image generation with the live data image,

we can encourage the synthetic image to capture regions of the scene also observed by the live data image, as shown in Sections 4 and 6. This concept is analogous to learning the mapping on the right of Figure 3, where, by using a pair of satellite and range sensor images as input, the regions of the scene to be present in the output synthetic image is no longer ambiguous, but constrained by the input range sensor image.

4 Self-Supervised Cross-Modality Localisation

Our localisation pipeline is composed of three steps: rotation inference, image generation, and pose estimation. We discuss them in detail in this section.

4.1 Rotation Inference

Given a paired map (e.g., satellite) image $A \in \mathcal{A}$ and live data (e.g., radar or lidar) image $B \in \mathcal{B}$ pre-scaled to have the same resolution but with an unknown $SE(2)$ offset, we seek to generate a synthetic image that contains the same appearance and observed scenes as B, but is pixel-wise aligned with A.

Let the $SE(2)$ pose difference between A and B be parametrised as $\begin{bmatrix} x & y & \theta \end{bmatrix}^T$, such that by rotating B by θ and then translating by $\begin{bmatrix} x & y \end{bmatrix}^T$, one can pixel-wise align B onto A. The image generation can be formulated as:

$$f(A, B) \rightarrow \tilde{B}_{\theta,\alpha}, \tag{1}$$

where $\alpha = \begin{bmatrix} x & y \end{bmatrix}^T$. $\tilde{B}_{\theta,\alpha}$ is a generated image of modality $\mathcal{B}$ that synthesises the input live sensor image B applied with a rotation of θ, followed by a translation of $\alpha = \begin{bmatrix} x & y \end{bmatrix}^T$. Thus, $\tilde{B}_{\theta,\alpha}$ is pixel-wise aligned with the input map image A, but contains the same observed scenes as B.

However, as originally noted by Tang et al. (2020b), the mapping in (1) is difficult to learn as the inputs A and B are offset by both a translation and a rotation. CNNs are inherently *equivariant* * to translation, but not to rotation (Lenc and Vedaldi 2015). As a result, the CNNs in the network cannot automatically utilise their mutual information and thereby capture their geometric relationship.

The method in Tang et al. (2020b) proposes to infer the rotation prior to image generation, namely, reducing (1) to two steps:

$$f_R(A, B) \rightarrow B_\theta, \tag{2}$$

$$f_G(A, B_\theta) \rightarrow \tilde{B}_{\theta,\alpha}. \tag{3}$$

Here f_R is a function that infers the rotation offset θ between A and B, and outputs B_θ, which is input image B rotated by θ. Now, B_θ is rotation-aligned with the map frame, and therefore offset with A only by a translation, which CNNs can naturally handle. f_G is an image generation function that produces the synthetic image $\tilde{B}_{\theta,\alpha}$ that is pixel-wise aligned with the input map image A. The experiments in Tang et al. (2020b) show that learning (2) and (3) sequentially resulted in better performance than learning (1) directly, as the former is congruous with the equivariance properties of CNNs.

In Tang et al. (2020b), the rotation inference function f_R is parametrised by a deep network as shown in Figure 6, where satellite imagery and radar images are used as an example. Given a coarse initial heading estimate, the live data image B is rotated a number of times with small increments to form a stack of rotated images $\{B\} = \{B_{\theta_0}, B_{\theta_1}, \ldots, B_{\theta_n}\}$, where the number of rotations n and the increment are design parameters. Each rotated image is further concatenated with the map image to form a stacked tensor input of n pairs of map and live data images. The network assigns a latent score per input pair, and outputs a softmaxed image from $\{B\}$ where the softmax weights are a function of learned latent scores:

$$f_R(A, B_{\theta_0}, B_{\theta_1}, \ldots, B_{\theta_n}) = \sum_i \frac{e^{z_i}}{\sum_i e^{z_i}} B_{\theta_i}, \tag{4}$$

where each z_i is the associated learned scalar latent score for the pair $\{A, B_{\theta_i}\}$.

A loss function enforces the output to correspond to B rotated to be rotation-aligned with A, namely B_θ. The core idea is that the network f_R will assign a large softmax weight to the image from $\{B\}$ whose heading most closely aligns with the map image A, and small weights to all other images in $\{B\}$.

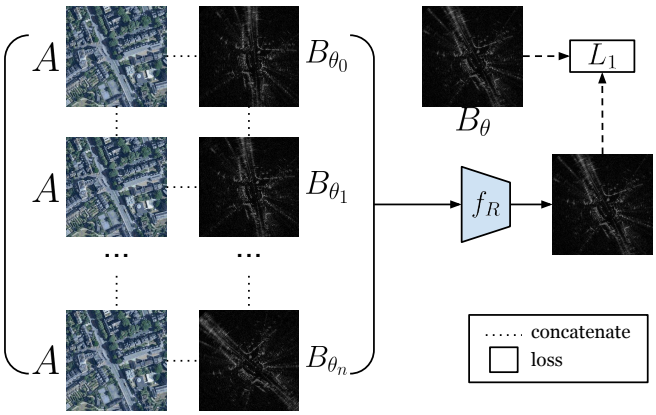


Figure 6. Prior work in Tang et al. (2020b) proposes a network to infer the rotation offset. The rotation offset is found by softmaxing an stack of rotated radar images to produce a radar image with the same heading as the satellite image.

If a metrically accurate heading ground truth θ is available, then one can rotate B to form a ground truth image target to B_θ used for supervising the rotation inference, as in Figure 6. In this work, however, we assume this is never the case, thus the network for f_R must learn to infer the rotation offset in a self-supervised fashion.

For this reason, while following the same architecture as Tang et al. (2020b), our method for inferring rotation uses a different training strategy that enables self-supervised learning. In order for the network f_R to produce the correct output, it must be able to infer the rotation from the solution space $\{B\}$, despite the modality difference between map image A and live data image B. We make the observation that if the network can infer the rotation offset from a stack of rotated live data images $\{B\}$, then, given some live data image B_{θ_i} with an arbitrary heading, f_R should also be able

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

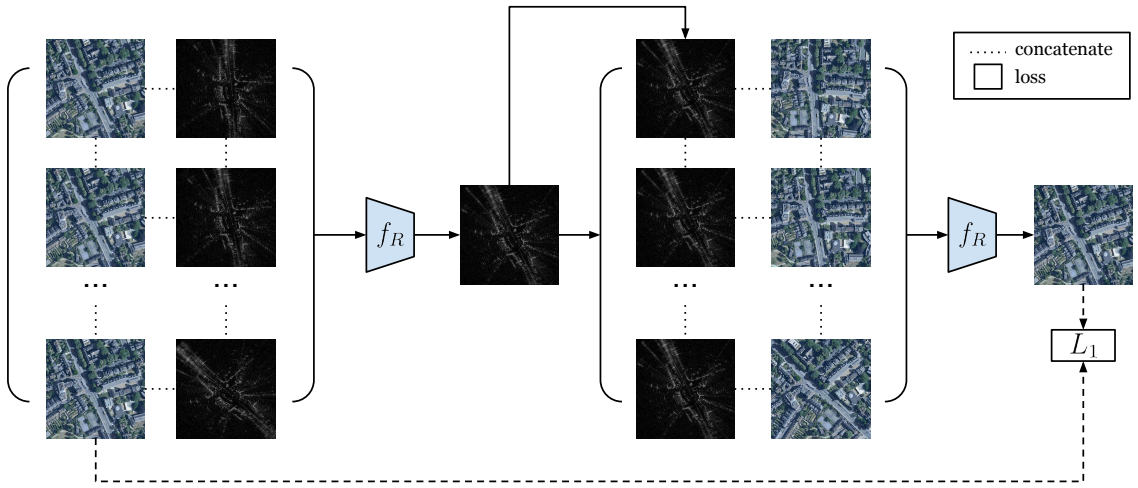


Figure 7. Given A and a rotation stack $\{B\}$ the network f_R finds B_{θ_i} by taking softmax. Then, given B_{θ_i} and a rotation stack $\{A\}$, the network outputs a softmaxed map image from $\{A\}$. A loss is applied to enforce the output of the second pass to be A, which in turn enforces the output of the first pass to be B_θ. Here both symbols for f_R in the figure refer to the same network with the same parameters, but at different forward passes.

to choose A_{θ_i} from a stack of rotated map images $\{A\}$, such that A_{θ_i} is rotation-aligned with B_{θ_i}. Then, if we have $B_{\theta_i} = B_\theta$, the softmaxed map image selected from $\{A\}$ should be the original, non-rotated image A, as A and B_θ are rotation-aligned.

As such, to learn rotation inference self-supervised, we need to pass through the network f_R twice. The first pass is identical as in the supervised approach in Figure 6, where we denote the output softmaxed image as B_{θ_i}. B_{θ_i} is then used as input to the second pass through network f_R, together with a stack of map images $\{A\} = \{A, A_{\phi_0}, A_{\phi_1}, \ldots, A_{\phi_m}\}$. The rotation angles $\begin{bmatrix}\phi_0 & \phi_1 & \ldots & \phi_m\end{bmatrix}$ can be chosen randomly, and the order of $\{A\}$ is shuffled such that the original non-rotated map image A can be at any index within $\{A\}$. Each image is concatenated with B_{θ_i}, the output of the first pass, to form the input stack for passing through f_R the second time. Note that the same network f_R is used in both passes.

The network is supervised with an L_1 loss that enforces the output of the second pass to be the non-rotated input map image A, which is an input in the first pass:

$$B_{\theta_i} = f_R(A, \{B\}), \tag{5}$$

$$\mathcal{L}_{L1}(f_R) = \mathbb{E}_{A,\{B\},B_{\theta_i}}[\|A - f_R(B_{\theta_i}, \{A\})\|_1], \tag{6}$$

where B_{θ_i} is the output of the first pass. Again, B_{θ_i} is what we are interested in solving as it uncovers the rotation offset between the input pair A and B, but we do not have a target for it. Minimizing the loss in (6) in turn enforces B_{θ_i} to be B_θ, as B_θ is rotation-aligned with A. Our approach is shown in Figure 7. We use an increment of 2° when forming the rotation stack $\{B\}$. We evaluate the effect of the rotation increment on solution error in Section 6.10 and justify our choice.

The estimate for the rotation offset, $\hat{\theta}$, can then be found from the arg-softmax for the rotation stack $\{B\}$.

4.2 Image Generation

Given a pair of rotation-aligned images A and B_θ we seek to generate a synthetic image $\tilde{B}_{\theta,\alpha}$ as in (3), where $\tilde{B}_{\theta,\alpha}$ is aligned with A in both rotation and translation. Tang et al. (2020b) learns the image generation function by a supervised approach, concatenating A and B_θ, and applying an encoder-decoder architecture, as shown in Figure 8. This is possible since a target for the synthetic image $\tilde{B}_{\theta,\alpha}$ can be obtained by applying the ground truth $SE(2)$ transform.

In the supervised approach in Tang et al. (2020b), a loss can be formed between the synthetic image and the target:

$$\mathcal{L}_{L1}(f_G) = \mathbb{E}_{A,B_\theta,B_{\theta,\alpha}}[\|B_{\theta,\alpha} - f_G(A, B_\theta)\|_1], \tag{7}$$

where $B_{\theta,\alpha}$ is the target image pixel-wise (rotation and translation) aligned with A, obtained with ground truth transform.

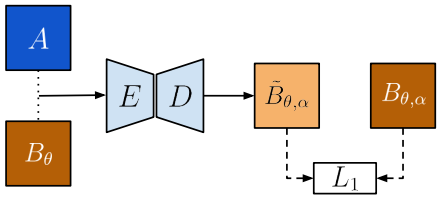


Figure 8. Architecture for image generation in prior supervised approach (Tang et al. 2020b).

To generate synthetic images self-supervised, we propose an architecture we call Pose-Aware Separable Encoder Decoder (PASED), shown in Figure 9. PASED is trained in two steps: the first is a pre-training, intra-modality process that can be supervised with ground truth image targets (top half of Figure 9), while the second handles cross-modality comparison (bottom half of Figure 9).

4.2.1 Pre-training Step Taking two random images B^1 and B^2 in the live data modality $\mathcal{B}$ from the training set, where B^1 and B^2 can be at arbitrary heading, we apply a known translation offset $\gamma \in \mathbb{R}^2$ to B^2. This forms an image B^2_γ that is a shifted version of B^2. We pass B^1 through an appearance encoder E_a that encodes its appearance and observed scenes. The shifted pair B^2_γ and B^2 is passed as inputs to a pose encoder E_p that encodes the translation offset between the input images. The latent spaces from the appearance encoder E_a and the pose encoder E_p are combined before passing through a decoder D, which outputs a synthetic image $\tilde{B}^1_\gamma$ that is B^1 shifted by a translation γ:

$$\tilde{B}^1_\gamma = D\big(E_a(B^1), E_p(B^2_\gamma, B^2)\big). \tag{8}$$

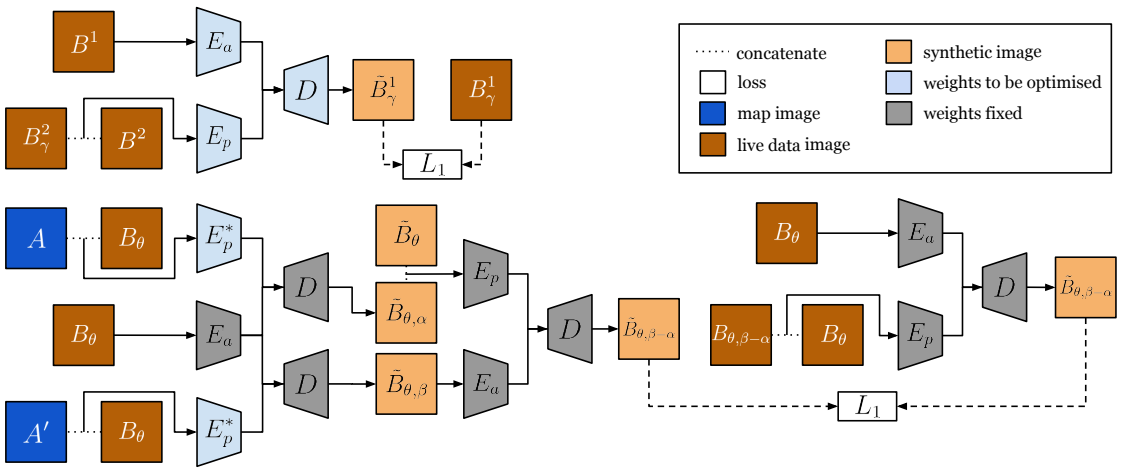


Figure 9. Top: during pre-training, we can learn an appearance encoder E_a, and a pose encoder E_p that discovers the translation offset between an image of $\mathcal{B}$ and a shifted version of itself. Bottom: Taking E_a, E_p, and D and fixing their weights, we seek to learn E_p^* which discovers the translation offset between two images from different modalities. E_a, E_p, and D can provide the necessary geometric and appearance relationships used for learning E_p^* self-supervised.

In other words, PASED discovers the translation offset between the two images passed as inputs to the pose encoder E_p, and applies the latent translation encoding to the input image to the appearance encoder E_a, for decoding using D. The pre-training can be supervised as γ is known, thus we can shift B^1 by γ to produce the target B^1_γ. The loss function can be formulated as

$$\mathcal{L}_{L1}(E_a, E_a, D) = \mathbb{E}_{B^1,B^2,B^1_\gamma,B^2_\gamma}[\left\| B^1_\gamma - \tilde{B}^1_\gamma \right\|_1], \quad (9)$$

and the parameters for networks E_a, E_p, and D can be optimised by minimizing (9).

The fact that we use different images B^1 and B^2 for inputs to E_a and E_p ensures appearance and pose are disentangled from each other. As shown later, this allows modules of PASED to be separated and re-combined with newly learned modules.

4.2.2 Cross-Modality Step In the second step, we fix the weights of E_a, E_p, and D which are optimised from the pre-training step. This narrows down the self-supervision problem to learning a cross-modality pose encoder E_p^* that discovers the translation offset between an image of modality $\mathcal{A}$ and another of $\mathcal{B}$. Taking a pair of cross-modal but rotation-aligned images A and B_θ as inputs, E_p^* should encode the unknown translation offset α between them. Concurrently, B_θ is fed to E_a to encode its appearance and the resulting latent space is combined with the latent space produced by the cross-modality pose encoder, $E_p^*(A, B_\theta)$, before being decoded by D. This encoder-decoder combination will generate a synthetic image $\tilde{B}_{\theta,\alpha}$ that should be pixel-wise (rotation and translation) aligned with A, but which we do not have a target for:

$$\tilde{B}_{\theta,\alpha} = D\big(E_a(B_\theta), E_p^*(A, B_\theta)\big). \quad (10)$$

We can apply a known shift to the centre position of A to query another map image A', where A' is offset with the rotation-aligned image B_θ by an unknown translation β. Using the same encoder-decoder combination as before, we can take A' and B_θ to generate a synthetic image $\tilde{B}_{\theta,\beta}$ that should be aligned with A' in both rotation and translation, which we also do not have a target for:

$$\tilde{B}_{\theta,\beta} = D\big(E_a(B_\theta), E_p^*(A', B_\theta)\big). \quad (11)$$

Furthermore, given B_θ and the networks learned from pre-training, we can easily generate $\tilde{B}_\theta$ by encoding a zero shift:

$$\tilde{B}_\theta = D\big(E_a(B_\theta), E_p(B_\theta, B_\theta)\big). \quad (12)$$

If we pass $\tilde{B}_\theta$ and $\tilde{B}_{\theta,\alpha}$ to the pre-trained pose encoder E_p, then the latent space will encode a shift of $-\alpha$. Combing this latent space with $E_a(\tilde{B}_{\theta,\beta})$, we can decode a synthetic image $\tilde{B}_{\theta,\beta-\alpha}$:

$$\tilde{B}_{\theta,\beta-\alpha} = D\big(E_a(\tilde{B}_{\theta,\beta}), E_p(\tilde{B}_\theta, \tilde{B}_{\theta,\alpha})\big). \quad (13)$$

Here $\beta - \alpha$ is a known value as it is the translation offset applied to A to obtain A'.

We can shift B_θ by $\beta - \alpha$ to get $B_{\theta,\beta-\alpha}$. Using $B_{\theta,\beta-\alpha}$ and B_θ, we can also generate $\tilde{B}_{\theta,\beta-\alpha}$ as a synthetic copy of $B_{\theta,\beta-\alpha}$, although now through a different way, using pre-trained networks E_a, E_p, and D, shown on the bottom right of Figure 9. Specifically, this can be expressed as

$$\tilde{B}_{\theta,\beta-\alpha} = D\big(E_a(B_\theta), E_p(B_{\theta,\beta-\alpha}, B_\theta)\big). \quad (14)$$

We can form $\tilde{B}_{\theta,\beta-\alpha}$ using two different combinations of inputs as in (13) and (14). Notably, $\tilde{B}_{\theta,\beta-\alpha}$ formed using (14) only passes through networks with weights optimised from the pre-training step and fixed during the cross-modality step, and therefore can be used as a target. A loss can then be established between the two synthetic images from (13) and (14) enforcing them to be the same, where the latter one is a target image:

$$\mathcal{L}_{L1}(E_p^*) = \mathbb{E}_{A,A',B_\theta,B_{\theta,\beta-\alpha}}[\|D\big(E_a(B_\theta), E_p(B_{\theta,\beta-\alpha}, B_\theta)\big) - D\big(E_a(\tilde{B}_{\theta,\beta}), E_p(\tilde{B}_\theta, \tilde{B}_{\theta,\alpha})\big)\|_1]. \quad (15)$$

By back-propagation, the loss in (15) optimises the network E_p^*, as $\tilde{B}_{\theta,\alpha}$ and $\tilde{B}_{\theta,\beta}$ are functions of E_p^*. Alternatively we can use the non-synthetic version $B_{\theta,\beta-\alpha}$ as the target, but in practice using the synthetic version $\tilde{B}_{\theta,\beta-\alpha}$ as in (14) led to faster convergence.

For the loss in (15) to be minimised, two conditions must hold true. First, the synthetic image $\tilde{B}_{\theta,\beta}$ must have correctly encoded the appearance and observed scenes in B_θ. Second, synthetic images $\tilde{B}_{\theta,\alpha}$ and $\tilde{B}_{\theta,\beta}$ must have the correct translations α and β, respectively. By satisfying these two constraints we can ensure the cross-modality pose encoder E_p^* is able to discover the translation offset across modalities, and is compatible with pre-trained networks E_a and D for image generation.

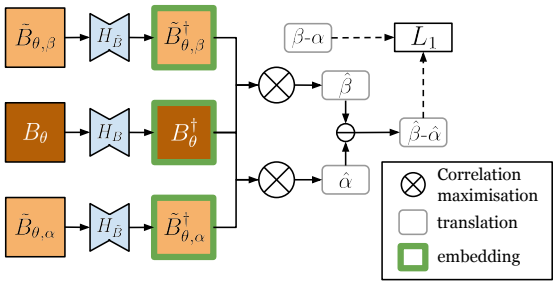


Figure 10. The networks H_B and $H_{\tilde{B}}$ are learned to project real live images and synthetic images to a joint embedding, where their translation offset can be found by maximising correlation.

4.3 Pose Estimation

Taking the pose-aligned synthetic image $\tilde{B}_{\theta,\alpha}$ and the rotation-aligned range sensor image B_θ, we embed them to a joint space, where their translation offset is found by maximising correlation on the learned embeddings. We denote the embedding network for real and synthetic images to be H_B and $H_{\tilde{B}}$, respectively, and the learned embeddings to be $B^{\dagger}_{\theta}$ and $\tilde{B}^{\dagger}_{\theta,\alpha}$:

$$B^{\dagger}_{\theta} = H_B(B_\theta), \quad \tilde{B}^{\dagger}_{\theta,\alpha} = H_{\tilde{B}}(\tilde{B}_{\theta,\alpha}). \tag{16}$$

The correlation maximisation is a parameter-free process that requires no additional learned modules, but is differentiable allowing gradients induced by the downstream loss to propagate to upstream learned modules. In this step, we can infer $\hat{\alpha} = \begin{bmatrix}\hat{x} & \hat{y}\end{bmatrix}^T$, which is our posterior estimate to the translation. Formally, this can be expressed as

$$\hat{\alpha} = \underset{p\in\mathbb{R}^2}{\arg\max}\, \tilde{B}^{\dagger}_{\theta,\alpha} \star B^{\dagger}_{\theta}, \tag{17}$$

where $\tilde{B}^{\dagger}_{\theta,\alpha} \star B^{\dagger}_{\theta}$ is the discrete cross correlation between $\tilde{B}^{\dagger}_{\theta,\alpha}$ and $B^{\dagger}_{\theta}$. This can be performed efficiently in the Fourier domain, as is done in prior works that use a similar approach (Barsan et al. 2018; Barnes et al. 2019; Tang et al. 2020b).

The embeddings are thus learned to further ensure the synthetic image and the live image can be correctly correlated. Without ground truth α, we can self-supervise using a similar approach as in learning PASED, by applying a known shift. The architecture for learning the embeddings is shown in Figure 10. Similar as in Section 4.2, $\tilde{B}_{\theta,\beta}$ can be obtained by shifting the map image A to get A'. Given learned deep embeddings $\tilde{B}^{\dagger}_{\theta,\beta}$ and $B^{\dagger}_{\theta}$, the translation offset by correlation maximisation is found to be $\hat{\beta}$:

$$\tilde{B}^{\dagger}_{\theta,\beta} = H_{\tilde{B}}(\tilde{B}_{\theta,\beta}), \tag{18}$$

$$\hat{\beta} = \underset{p\in\mathbb{R}^2}{\arg\max}\, \tilde{B}^{\dagger}_{\theta,\beta} \star B^{\dagger}_{\theta}. \tag{19}$$

The difference of the two offsets $\beta - \alpha$ is known, and can be used to establish a loss term:

$$\mathcal{L}_{L1}(H_B, H_{\tilde{B}}) = \mathbb{E}_{B_\theta,\tilde{B}_{\theta,\alpha},\tilde{B}_{\theta,\beta}}[\left\|\beta - \alpha - (\hat{\beta} - \hat{\alpha})\right\|_1]. \tag{20}$$

The overall pipeline for data flow at inference time is shown in Figure 11.

5 Implementation Details

Here we provide details on the architecture of the various networks used in our method, and the associated hyper-parameters. We make use of the following abbreviations:

- RP(p): 2D reflection padding of p
- Conv($C_{\text{in}}, C_{\text{out}}, k, s, p$): convolution with C_{in} input channels, C_{out} output channels, kernel size k, stride s, padding p, and bias
- IN: instance normalisation
- ReLU: rectified linear unit
- LReLU(m): leaky ReLU with negative slope m
- Drop(d): dropout with ratio d
- ConvT($C_{\text{in}}, C_{\text{out}}, k, s, p, p_{\text{out}}$): transposed convolution with C_{in} input channels, C_{out} output channels, kernel size k, stride s, padding p, output padding p_{out}, and bias

Table 1. Architecture for rotation inference

Rotation Inference Function f_R
Input shape - $n \times 4 \times 256 \times 256$ where $C = 4$, $H = W = 256$
Conv(4, 32, 3, 2, 1) + IN + ReLU
Conv(32, 64, 3, 2, 1) + IN + ReLU
Conv(64, 128, 3, 2, 1) + IN + ReLU
Conv(128, 256, 3, 2, 1) + IN + ReLU
Latent shape - $n \times 256 \times 16 \times 16$
Take the mean along C, H, W + Softmax + Reshape
Latent vector shape - $1 \times n$, which are the softmax weights
Matrix-multiple softmax weights with the input
Shape of the multiplication product - $4 \times 256 \times 256$
Extract the associated channel(s) to get B_θ (or A_{θ_i} during training)

The network architectures are shown in Tables 1 to 4. For comparison against the prior supervised approach, we use the same architectures where possible. We implemented the image generation network for the prior supervised approach to have the same latent space size at the bottleneck, and the same number of down-sample and up-sample layers as in our method.

Our method is implemented in PyTorch (Paszke et al. 2019). For training rotation inference f_R and networks for image generation E_a, E_p, E_p^*, and D, we use a learning rate of 2e^{-4}. For learning the embedding networks H_B and $H_{\tilde{B}}$, we use a learning rate of 2e^{-6}. We use Adam (Kingma and Ba 2015) as the optimiser for all experiments. The training is terminated when the validation loss increases for more than 5 epochs, or reaching a maximum number of epochs. This results in 80 to 150 epochs of training for learning f_R, E_a, E_p, E_p^*, and D, depending on the dataset and the specific experiment, and 10 to 20 epochs for learning H_B and $H_{\tilde{B}}$. The inference runs at about 10 Hz on a single 1080

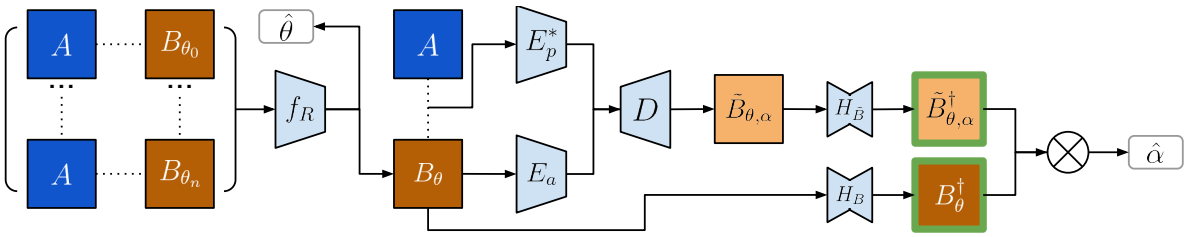


Figure 11. Overall data flow of our method at inference: given map image A and live data image B, based on the initial heading estimate, we form a stack of rotated images $\{B_{\theta_0}, \ldots, B_{\theta_n}\}$, from which f_R discovers B_θ that is B rotated to be rotation-aligned with A. This process also infers the heading estimate $\hat{\theta}$. A and B_θ are used to generate a synthetic image $\tilde{B}_{\theta,\alpha}$ that has the same appearance and observed scene as B_θ and is pose-aligned with A. $\tilde{B}_{\theta,\alpha}$ and B_θ are projected to deep embeddings $\tilde{B}^\dagger_{\theta,\alpha}$ and $B^\dagger_\theta$, where the estimate for the translation offset $\hat{\alpha}$ is found by correlation maximisation.

Table 2. Architecture of for image generation

Appearance Encoder E_a
RP(3) + Conv(1, 16, 7, 1, 0) + IN + ReLU
Conv(16, 32, 3, 2, 1) + IN + ReLU
Conv(32, 64, 3, 2, 1) + IN + ReLU
Conv(64, 128, 3, 2, 1) + IN + ReLU
Conv(128, 256, 3, 2, 1) + IN + ReLU
ResNet blocks ($\times 9$):
Conv(256, 256, 3, 1, 0) + IN + ReLU + Drop(0.5)
Conv(256, 256, 3, 1, 0) + IN
Intra-Modality Pose Encoder E_p
RP(3) + Conv(2, 16, 7, 1, 0) + IN + ReLU
Conv(16, 32, 3, 2, 1) + IN + ReLU
Conv(32, 64, 3, 2, 1) + IN + ReLU
Conv(64, 128, 3, 2, 1) + IN + ReLU
Conv(128, 256, 3, 2, 1) + IN + ReLU
ResNet blocks ($\times 9$):
Conv(256, 256, 3, 1, 0) + IN + ReLU + Drop(0.5)
Conv(256, 256, 3, 1, 0) + IN

Cross-Modality Pose Encoder E_p^*
RP(3) + Conv(4, 16, 7, 1, 0) + IN + ReLU
Conv(16, 32, 3, 2, 1) + IN + ReLU
Conv(32, 64, 3, 2, 1) + IN + ReLU
Conv(64, 128, 3, 2, 1) + IN + ReLU
Conv(128, 256, 3, 2, 1) + IN + ReLU
ResNet blocks ($\times 9$):
Conv(256, 256, 3, 1, 0) + IN + ReLU + Drop(0.5)
Conv(256, 256, 3, 1, 0) + IN
Decoder D
ConvT(512, 256, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
ConvT(256, 128, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
ConvT(128, 64, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
ConvT(64, 32, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
RP(3) + Conv(32, 1, 7, 1, 0) + Sigmoid

Table 3. Image generation for our implementation of RSL-Net [36] for comparison

Encoder E
RP(3) + Conv(4, 32, 7, 1, 0) + IN + ReLU
Conv(32, 64, 3, 2, 1) + IN + ReLU
Conv(64, 128, 3, 2, 1) + IN + ReLU
Conv(128, 256, 3, 2, 1) + IN + ReLU
Conv(256, 512, 3, 2, 1) + IN + ReLU
ResNet blocks ($\times 9$):
Conv(512, 512, 3, 1, 0) + IN + ReLU + Drop(0.5)
Conv(512, 512, 3, 1, 0) + IN
Decoder D
ConvT(512, 256, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
ConvT(256, 128, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
ConvT(128, 64, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
ConvT(64, 32, 3, 2, 1, 1) + IN + ReLU + Drop(0.5)
RP(3) + Conv(32, 1, 7, 1, 0) + Sigmoid

Table 4. U-Net architecture for learning embeddings

Embedding Networks H_B and $H_{\tilde{B}}$
Conv(1, 32, 4, 2, 0)
LReLU(0.2) + Conv(32, 64, 4, 2, 0) + IN
LReLU(0.2) + Conv(64, 128, 4, 2, 0) + IN
LReLU(0.2) + Conv(128, 256, 4, 2, 0) + IN
LReLU(0.2) + Conv(256, 512, 4, 2, 0) + IN
LReLU(0.2) + ReLU + Conv(512, 1024, 4, 2, 0)
ReLU + ConvT(1024, 512, 4, 2, 1, 0) + IN
ReLU + ConvT(512, 256, 4, 2, 1, 0) + IN
ReLU + ConvT(256, 128, 4, 2, 1, 0) + IN
ReLU + ConvT(128, 64, 4, 2, 1, 0) + IN
ReLU + ConvT(64, 32, 4, 2, 1, 0) + IN
ReLU + ConvT(32, 1, 4, 2, 1, 0) + Sigmoid
With skip connections in-between intermediate layers

Ti GPU. We use a batch size of 32 for all experiments unless otherwise stated.

6 Experimental Validation

We evaluate on several public, real-world datasets collected with vehicles equipped with on-board range sensors. The datasets we use come with metric ground truths that are decently accurate, though we noticed the GPS/INS solutions in certain places can drift up to a few metres.

We add large artificial pose offsets to the ground truth when querying for a satellite image, thereby simulating a realistic robot navigation scenario where the initial pose estimate can solve place recognition, but is too coarse for the robot's metric pose. Using a map (e.g., satellite) image queried at this coarse initial pose estimate, our method solves metric localisation by comparing against the live sensor data. The true pose offsets are hidden during training as our method is self-supervised, and are only revealed at test time for evaluation purposes.

The artificial offset is chosen such that the initial estimate has an unknown heading error in the range $[-\frac{\pi}{8}, \frac{\pi}{8}]$, therefore given the initial estimate θ_0, the rotation inference must choose a solution space of at least $[\theta_0 - \frac{\pi}{8}, \theta_0 + \frac{\pi}{8}]$ to guarantee the correct solution can be found. We use a pixel-wise translation error in the range $[-25, 25]$ pixels. Depending on the resolution for a specific experiment, this corresponds to an error of at least $[-10\text{m}, 10\text{m}]$ and up to

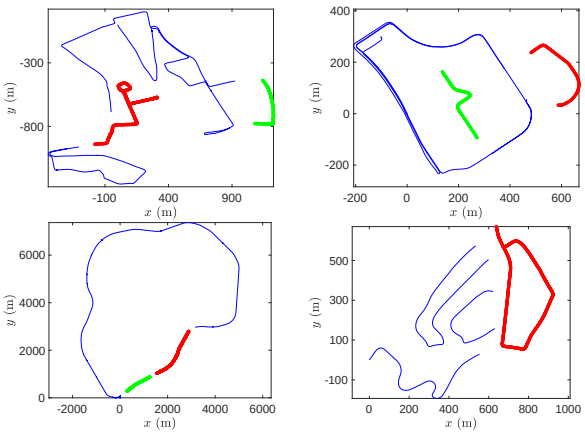


Figure 12. Training (blue), validation (green), and test (red) trajectories for RobotCar (top left), `KAIST` (top right), `Sejong` (bottom left) and `20111003_drive0034` (bottom right). Certain data are removed to avoid overlap between the splits.

more than $[-20\text{m}, 20\text{m}]$. All experiments use input images of size 256×256.

6.1 Radar Localisation Against Satellite Imagery

We evaluate our method on two datasets with FMCW radar and GPS: the Oxford Radar RobotCar Dataset (Barnes et al. 2020) and the MulRan Dataset (Kim et al. 2020). The satellite images for RobotCar are queried using Google Maps Platform [†]. For MulRan they are queried using Bing Maps Platform [‡], as high-definition Google satellite imagery is unavailable at the place of interest.

The GPS/INS and range sensor data for all datasets used are timestamped. To create the ground truth, for each frame of range sensor data, we find its associated latitude, longitude, and heading from the GPS/INS data based on its timestamp, and query a satellite image with the corresponding latitude and longitude; we then rotate the range sensor image with the ground truth heading to generate a rotation-aligned range-sensor image. To simulate a coarse initial estimate, we shift the centre of the satellite images and rotate the range sensor image by randomly-sampled translation and rotation offset respectively when forming the range sensor - satellite pairs.

We benchmark against the prior supervised method RSL-Net (Tang et al. 2020b) in our experiments, which is evaluated only on the RobotCar Dataset. Both datasets contain repeated traversals of the same routes. We separately train, validate, and test for every dataset, splitting the data as in Figure 12. For the RobotCar Dataset, we split the trajectories the same way as in Tang et al. (2020b) for a fair comparison. For the RobotCar Dataset, the training set consists of training data from sequences no. 2, no. 5, and no. 6, while we test on the test data from sequence no. 2. For the MulRan Dataset, we used sequences `KAIST 01` and `Sejong 01`. The resulting test sets feature an urban environment (RobotCar), a campus (`KAIST 01`) and a highway (`Sejong 01`).

We test on every fifth frame, resulting in 201 frames from the RobotCar Dataset and 358 from the MulRan Dataset, spanning a total distance of near $4\,\text{km}$. The resolution used is 0.8665 m/pixel for RobotCar and 0.7876 m/pixel for MulRan. All sensor data are pre-scaled to have the same resolution as satellite images. The mean errors and standard deviations around the mean are reported in Table 5. Starting from a large initial offset, we can localise to an error of a few metres in translation and a few degrees in heading. Our method achieves an error on par with the supervised approach in Tang et al. (2020b), while requiring no metrically accurate ground truth for training.

6.2 Lidar Localisation Against Satellite Imagery

For this experiment, we evaluate on the RobotCar Dataset (Barnes et al. 2020) which also has two Velodyne HDL-32E lidars mounted in a tilted configuration, and KITTI (raw dataset, Geiger et al. (2013)) which has a Velodyne HDL-64E lidar and GPS data.

For the RobotCar Dataset, the trajectories are split into training, validation, and test sets approximately the same way as in Section 6.1. For the KITTI Dataset, the training set includes sequences `20110929_drive0071`, `20110930_drive0028`, and `20111003_drive0027`. Sequence `20110926_drive0117` is used for validation. Finally, data in `20111003_drive0034` are split into training and test, as shown in Figure 12. To turn 3D lidar point-clouds to lidar images, the point-clouds are projected to the x-y plane. We discard points with z values smaller than zero to remove ground points when creating the lidar images.

Since lidars have a shorter range than radars, we use satellite images of a greater zoom level, with resolution 0.4332 m/pixel for RobotCar and 0.4592 m/pixel for KITTI. The test set consists of 200 frames for RobotCar and 253 for KITTI, spanning a total distance of near $3\,\text{km}$. The test set for KITTI features a residential area. The results are reported in Table 6. The error on the RobotCar Dataset is smaller when using lidar for localisation than when using radar.

6.3 Radar Localisation Against Prior Lidar Map

Though our method is designed for localising against satellite imagery, we show it can also handle more standard forms of cross-modality localisation. Here we build a lidar map using a prior traversal, and localise using radar from a later traversal.

We demonstrate on the RobotCar and MulRan datasets, where we use the same resolution as in Section 6.1. For RobotCar, we use ground truth to build a lidar map from sequence no. 2. Radar data in the training sections from no. 5 and no. 6 as in Figure 12 form the training set, while the test section from sequence no. 5 forms the test set. For MulRan, lidar maps are built from `KAIST 01` and `Sejong 01`, and we localised using radar data from `KAIST 02` and `Sejong 02`, which are split into training, validation, and test sets. This resulted in a test set consisting of 201 frames from RobotCar and 272 frames from MulRan, spanning a total

[†] https://developers.google.com/maps/documentation/maps-static/intro/
[‡] https://docs.microsoft.com/en-us/bingmaps/

Table 5. Mean error and error standard deviation for radar localisation against satellite imagery.

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar (ours)	3.44	5.40	**3.03**	3.97	6.23	4.44	7.50	**3.04**
RobotCar (RSL-Net (Tang et al. 2020b), supervised)	**2.74**	**4.26**	3.12	**3.16**	**4.92**	**3.93**	**6.32**	3.25
MulRan (ours)	6.02	**7.02**	2.92	7.64	**8.91**	**5.75**	6.87	3.64
MulRan (RSL-Net)	**5.85**	7.11	**1.88**	**7.42**	9.03	6.50	**6.46**	**2.41**

Table 6. Mean error and error standard deviation for lidar localisation against satellite imagery.

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar (ours)	**1.54**	**1.85**	2.29	**3.55**	**4.27**	**1.54**	**2.29**	3.10
RobotCar (RSL-Net)	2.31	2.55	**2.08**	5.33	5.89	2.26	2.57	**2.91**
KITTI (ours)	3.05	3.13	1.67	6.64	6.82	**3.59**	**3.78**	2.57
KITTI (RSL-Net)	**2.45**	**2.79**	**1.59**	**5.34**	**6.08**	3.70	3.88	**2.22**

Table 7. Mean error and error standard deviation for radar localisation against prior lidar map.

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar (ours)	**2.21**	**2.57**	2.65	**2.55**	**2.97**	**2.30**	3.53	**2.06**
RobotCar (RSL-Net)	2.66	3.41	**2.45**	3.07	3.93	2.85	**3.11**	2.18
RobotCar (CycleGAN)	6.41	9.05	2.65	7.40	10.44	7.43	7.17	2.06
MulRan (ours)	3.57	3.26	2.15	4.53	4.13	4.29	4.84	2.38
MulRan (RSL-Net)	**3.37**	**2.61**	**1.40**	**4.28**	**3.32**	**3.38**	**4.12**	**1.71**
MulRan (CycleGAN)	4.84	4.39	2.15	6.14	5.58	5.78	4.96	2.38

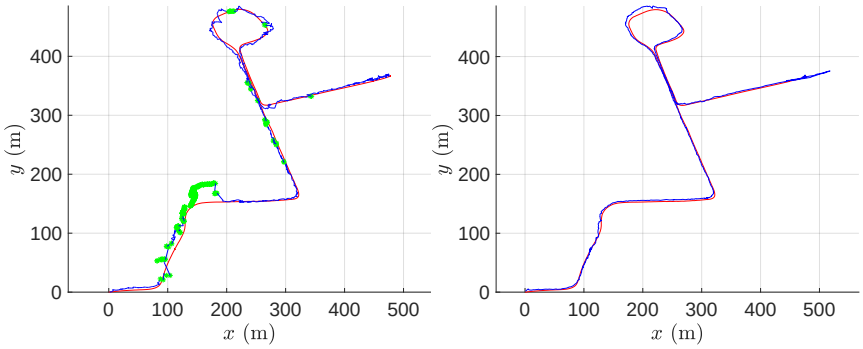


Figure 13. Estimated pose (blue) vs. ground truth pose (red) for localising a radar (left) and a lidar (right) against satellite imagery. Our system tracks the vehicle's pose over 1 km, where we occasionally fall back to odometry for the radar experiment (green). Our system is stand-alone and requires GPS only for the first frame.

distance of near $4\,\mathrm{km}$. The localisation results are shown in Table 7.

This experiment is more suitable for naïve image generation methods such as CycleGAN (Zhu et al. 2017) than previous experiments, since the field-of-view is considerably more compatible as both modalities are from range sensors. In Table 7, we show results where we replaced the image generation stage of our method by CycleGAN while keeping other modules unchanged. The localisation results are however much worse when modality $\mathcal{A}$ is satellite imagery, as shown qualitatively in Figure 5.

6.4 Online Pose-Tracking System

In prior experiments we assumed place recognition is always available, providing a coarse initial estimate for every frame. Here we present a stand-alone pose-tracking system by continuously localising against satellite imagery. Given a coarse initial estimate (e.g., from GPS) for the first frame, the vehicle localises and computes its pose within the satellite map. The initial estimate for every frame onward is then set to be the computed pose of the previous frame. We only need place recognition once at the very beginning; the vehicle then tracks its pose onward without relying on any other measurements.

6.4.1 Introspection As localising using satellite imagery is challenging, the result will not always be accurate. Our method, however, naturally allows for introspection. A synthetic image $\tilde{B}_{\theta,\alpha}$ was generated from A and B_θ. We can apply a known small translation offset δ to A to form A_δ. Taking A_δ and B_θ we can generate $\tilde{B}_{\theta,\alpha+\delta}$. Finally, we can compute a translation offset $\hat{\delta}$ by passing $\tilde{B}_{\theta,\alpha+\delta}$ and $\tilde{B}_{\theta,\alpha}$ through the learned embeddings and maximising correlation:

$$\hat{\delta} = \underset{p\in\mathbb{R}^2}{\arg\max}\, \tilde{B}^{\dagger}_{\theta,\alpha+\delta} \star \tilde{B}^{\dagger}_{\theta,\alpha} \tag{21}$$

Let $d_{\text{intro}} = \left\| \delta - \hat{\delta} \right\|$. A large value of d_{intro} indicates the generated images are erroneous. This allows us to examine the solution quality; our system falls back to using odometry for dead-reckoning when d_{intro} exceeds a threshold. We do not require high-quality odometry, but rather only use a naïve approach by directly maximising correlation between two consecutive frames without any learned modules. In our experiments, we set δ to be $\begin{bmatrix}10 & 10\end{bmatrix}^T$, and d_{intro} to be 5.

Table 8. Ablation study for using reduced training data, evaluated on radar localisation against satellite imagery on the RobotCar Dataset.

	Mean Error (metric)			Mean Error (pixel)	
	x (m)	y (m)	θ (°)	x	y
RobotCar (full)	**3.44**	**5.40**	**3.03**	**3.97**	**6.23**
RobotCar (first 20%)	7.96	7.45	6.03	9.18	8.59
RobotCar (every 10th)	4.36	6.18	4.40	5.03	7.14

Table 9. Mean error for using various choices of network width and depth, evaluated on lidar localisation against satellite imagery on the RobotCar Dataset.

	Mean Error (metric)			Mean Error (pixel)		Number of Parameters
	x (m)	y (m)	θ (°)	x	y	(M)
RobotCar, channels $\times\frac{1}{2}$	1.57	1.93	2.58	3.62	4.45	5.23
RobotCar, ours	**1.54**	1.85	2.29	**3.55**	4.27	20.86
RobotCar, channels $\times 2$	1.58	**1.65**	**2.27**	3.64	**3.80**	83.37
RobotCar, depth -1	1.70	2.09	2.47	3.92	4.82	5.22
RobotCar, ours	**1.54**	**1.85**	**2.29**	**3.55**	**4.27**	20.86
RobotCar, depth $+1$	1.56	1.98	2.39	3.60	4.57	83.41

Table 10. Mean error and error standard deviation for using refined resolution, evaluated for radar localisation against satellite imagery. We also include results for a hybrid approach utilising both resolutions.

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar, 0.8665 m/pixel	3.44	5.40	**3.03**	**3.97**	6.23	4.44	7.50	**3.04**
RobotCar, 0.4332 m/pixel	**2.23**	**2.42**	3.22	5.29	**5.74**	**2.49**	**3.02**	3.13
RobotCar, hybrid	2.56	3.60	3.03	5.91	8.32	3.78	5.45	3.04

6.4.2 Results We conduct two experiments on the test set of RobotCar, one where we track a radar using satellite imagery, and one where we track a lidar. For both experiments we run localisation at $4\,\text{Hz}$. The results are shown in Figure 13. If the solution error is too large, then the initial estimate will be too off for a sufficient overlap between the next queried satellite image and live data, resulting in losing track of the vehicle. Although the solution error is large for certain frames, our system continuously localises the vehicle for over a kilometre without completely losing track. For the lidar experiment, the solutions are sufficiently accurate to not require any odometry. Each solution only uses a single frame of data (plus the solution from the previous frame for the initial estimate), and we make no attempt at windowed/batch optimisation or loop closures.

6.5 Ablation Study

We perform an ablation study to investigate the effect of reduced training data. For radar localisation against satellite imagery on the RobotCar Dataset, we trained a model using approximately the first $20\,\%$ of training data, and another using every 10th frame of training data. The results are shown in Table 8.

Noticeably, the choice of selecting uniformly distributed data in contrast to utilising only the first $20\,\%$ leads to a more varied dataset, and as such achieves better performances despite the lower number of samples.

6.6 Testing on Different Sequences

For the results in Tables 5, 6, and 7, the training and test data, though with zero spatial overlap, are from the same sequences. The RobotCar and MulRan datasets contain repeated sequences of the same trajectory. Here we show results where the test set trajectories are entirely from different sequences, to demonstrate the capability of generalising not only to unseen places, but also to range sensor data recorded on different days as the training data.

We arbitrarily selected sequences no. 7 and no. 15 as the new test sequences for RobotCar, and sequences `KAIST 02` and `Sejong 02` for MulRan. The same test trajectories as in Figure 12 are extracted for the new test sequences. The training trajectories and sequences remain unchanged.

The localisation results are shown in Table 11 and Table 12. The mean errors are fairly consistent across different sequences.

6.7 Circular Initial Offset

For a fair comparison against the supervised approach in (Tang et al. 2020b), we assume the initial offset for both x and y is uniformly distributed and in the range $[-25, 25]$ pixels, which was also employed in (Tang et al. 2020b). However, a more realistic initial offset that more closely mimics the error characteristics of a GPS would be sampling the initial offset from a circular area of radius r from the ground truth position.

Without re-training, we evaluate the model performance where the initial translation offset is sampled uniformly from a circular area with a radius of 25 pixels, and centred at the ground truth position, while the sampling for the initial heading offset remains unchanged. The localisation results are summarised in Tables 13 and 14.

6.8 Trade-offs on Network Width and Depth

Here we show the effects of network width (number of channels in each layer) and depth (number of layers) on solution quality and the associated trade-offs, and justify choosing the architecture shown in Tables 1 to 4. In this experiment, we vary the width or depth of networks for rotation inference and image generation, namely f_R, E_a, E_p, E_p^*, D, while keeping H_B and $H_{\tilde{B}}$ unchanged. The results are shown in Table 9.

First, we fix the network depth while halving or doubling the number of channels in each layer. When the width is reduced, the network representation power is noticeably

Table 11. Results for radar localisation against satellite imagery on multiple test sequences

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar (sequence no. 1)	3.44	5.40	3.03	3.97	6.23	4.44	7.50	3.04
RobotCar (sequence no. 7)	3.44	5.15	3.96	3.97	5.95	5.76	5.44	4.32
RobotCar (sequence no. 15)	3.21	5.21	3.80	3.70	6.01	2.83	5.76	4.20
MulRan (`KAIST01, Sejong 01`)	6.02	7.02	2.92	7.64	8.91	5.75	6.87	3.64
MulRan (`KAIST02, Sejong 02`)	6.44	6.59	5.15	8.19	8.37	6.64	6.62	7.32

Table 12. Results for lidar localisation against satellite imagery on multiple test sequences

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar (sequence no. 1)	1.54	1.85	2.29	3.55	4.27	1.54	2.29	3.10
RobotCar (sequence no. 7)	1.38	1.72	2.35	3.18	3.98	1.37	1.83	2.71
RobotCar (sequence no. 15)	1.67	1.81	2.05	3.87	4.17	1.42	1.84	2.12

Table 13. Results for radar localisation against satellite imagery using circular translation offset

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar (ours)	2.99	4.20	2.80	3.45	4.84	2.46	4.55	2.48
RobotCar (RSL-Net)								
MulRan (ours)	6.28	**6.65**	3.22	7.98	**8.44**	5.32	5.32	3.88
MulRan (RSL-Net)	**5.80**	8.03	**1.90**	**7.36**	10.20	**4.82**	**5.01**	**2.90**

Table 14. Results for lidar localisation against satellite imagery using circular translation offset

	Mean Error (metric)			Mean Error (pixel)		Error STD (metric)		
	x (m)	y (m)	θ (°)	x	y	x (m)	y (m)	θ (°)
RobotCar (ours)	**1.41**	**1.78**	1.84	**3.26**	**4.10**	**0.99**	**1.50**	2.45
RobotCar (RSL-Net)	2.45	2.28	**1.31**	5.66	5.26	2.71	2.02	**1.37**
KITTI (ours)	3.09	3.11	1.18	6.72	6.79	**2.73**	2.88	1.31
KITTI (RSL-Net)	**2.48**	**2.29**	**1.03**	**5.40**	**4.99**	3.58	**2.54**	**0.98**

affected, indicated by an increase in solution error in all of x, y, and θ. When the width is doubled, we achieved a slight reduction on the rotation error and overall translation error. However, the total number of network parameters increase by a factor of 4 when the width is doubled, greatly limiting the maximum affordable batch size, and thereby increasing the training time. As a result, we opted the architecture presented in Section 5.

Next, we keep the number of channels the same in the first layer, and study the effect of making the networks shallower or deeper (by one layer). For image generation we use an encoder-decoder architecture with stride 2 in the convolution layers (after the first layer) as shown in Table 2, thus the height and width of the representation decrease by a factor of 2 after each layer, becoming 16×16 at the bottleneck with an input of size 256×256. Intuitively, a trade-off exists such that deeper networks theoretically have stronger representation power, but will result in reduced representation size at the bottleneck, making image reconstruction harder. The results in Table 9 suggest that our choice of network depth is optimal with noticeably smaller translation error, which is affected by the quality of image generation.

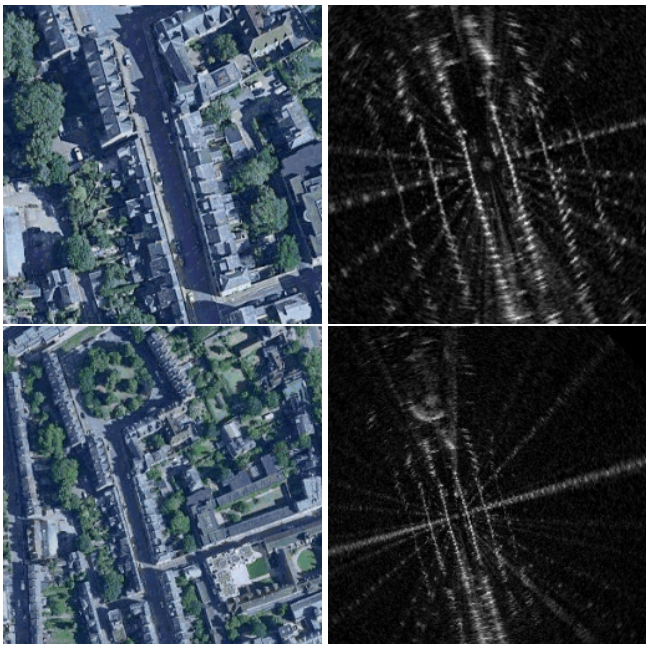


Figure 14. A pair of satellite and radar images queried using zoom levels of 18 (top) vs. 17 (bottom) at roughly the same centre position. Although "zooming-in" can lead to a more refined resolution, certain regions far away are not seen in the resulting radar images, despite being observed by the sensor.

6.9 Choice of Image Resolution

In the results presented in Sections 6.1 and 6.2, we used a resolution of 0.8665 m/pixel for experiments on radar localisation against satellite imagery for the RobotCar Dataset, and a resolution of 0.4332 m/pixel for lidar localisation against satellite imagery. The resolutions used correspond to a "zoom level" of 17 and 18 respectively when querying from Google Maps Platform, and are chosen based on the effective sensing range of radar and lidar.

To study the effect of image resolution on solution quality, we created another dataset for radar localisation against

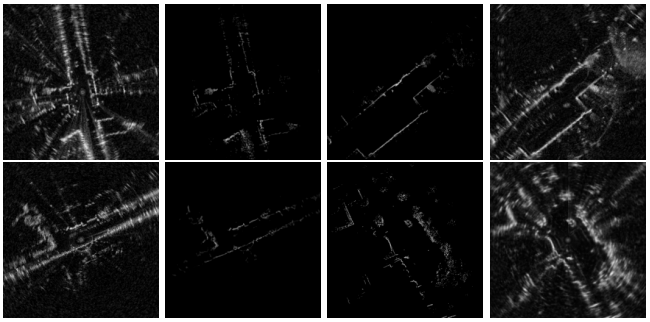


Figure 15. Qualitative results of CycleGAN for domain adaptation between a single scan of radar and lidar data. From left to right: a real radar image and its synthetic lidar image, a real lidar image and its synthetic radar image. Top: CycleGAN applied in Cartesian coordinate representation. Bottom: CycleGAN applied in polar coordinate representation.

satellite imagery from the RobotCar Dataset, where the resolution is set to be 0.4332 m/pixel, effectively "zooming in" on both the radar and satellite images. A comparison between images with the refined resolution and images used in the experiment in Section 6.1 is shown in Figure 14. We consider the same translation error in pixels for the initial estimate when comparing the two resolutions. The localisation results are shown in Table 10.

Overall, choosing a resolution of 0.4332 m/pixel resulted in slightly larger mean error in x and θ. Although zoomed-in images offer more refined resolution, without changing the size of the input images, by "zooming-in", measurements far from the sensor are essentially discarded, as shown in Figure 14, thereby limiting the amount of information being passed through the network.

Due to the pixel-based nature of CNNs, the upper limit our method can handle in terms of initial translation offset is also inherently in pixels. Without changing the initial pose offset expressed in pixels, models trained with higher resolution images offer a reduction in metric error, at the cost, however, of needing a smaller metric initial offset. In real-world applications, the coarse initial offset may be large, for example in places where direct GPS signals are occluded, and as such models trained with lower resolution images are needed to handle this large initial offset. For a smaller metric initial offset, a model trained with higher resolution images can be used to provide a more refined pose estimate.

With this in mind, we present a "hybrid" approach that ustilises both lower and higher resolution images. Specifically, given an initial offset in the range $[-21.66\text{m}, 21.66\text{m}]$ which corresponds to $[-25, 25]$ pixels with a resolution of 0.8665 m/pixel, we first compute a solution using a model trained with lower resolution images. Next, we use the estimated pose as the new initial translation offset and compute a refined solution using a model trained with higher resolution – at 0.4332 m/pixel. This "hybrid" approach allows for the best of both worlds: the lower resolution model allows large metric initial offsets, while the higher resolution model provides a further refinement, reducing the metric error. The results are shown in the last row of Table 10: the metric error in both x and y are reduced compared to using a lower resolution model only by the sequential refinement with a higher resolution model.

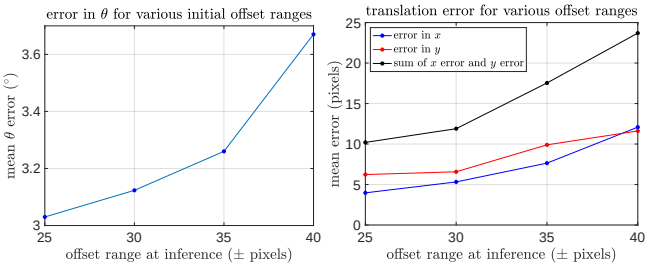


Figure 16. Mean error in rotation (left) and translation (right) with larger initial translation offset range, evaluated for radar localisation against satellite imagery on the RobotCar Dataset.

Table 15. Mean θ error and error standard deviation for various rotation increments, evaluated for radar localisation against satellite imagery on the RobotCar Dataset.

Rotation Increment (°)	Mean Error in θ (°)	**Error STD (°)**
1	3.29	3.91
2	**3.03**	**3.04**
3	4.04	4.93
4	4.26	5.86

6.10 Choice of Rotation Increment

We have chosen a rotation increment of $2°$ when forming the rotation stack $\{B\}$ for all of our experiments, here we justify this choice by comparing the resulting rotation error for various increments.

By intuition, the solution error on θ should decrease with finer increments. However, a trade-off exists such that for the same range of initial heading error ($[-\frac{\pi}{8}, \frac{\pi}{8}]$ in all our experiments), the number of rotated images that form $\{B\}$, n, increases due to the smaller rotation increments. This in turn increases the memory usage during training. When training on 2 1080 Ti GPUs, limited by the memory requirement, we could only afford a batch size of 16 when the increment is set to $1°$, while we can use a batch size of at least 32 when the increment is $2°$ or larger.

Shown in Table 15, the mean error in θ when choosing $1°$ as the rotation increment is slightly larger than when using $2°$, primarily because of the fact that the batch size is too small, in our hypothesis. We used a batch size of 32 for the experiments where the increment is $2°$, $3°$, and $4°$, and it is clear that the solution error increases with larger increments and thus coarser sampling. For an increment of $4°$ we could also afford a batch size of 64, however the resulting θ error was higher than using 32 as the batch size.

6.11 Handling Larger Initial Offset

So far all experiments presented start from an initial translation offset of $[-25, 25]$ pixels, which corresponds to more than $[-20\text{m}, 20\text{m}]$ for the radar experiments in Section 6.1. In practice, the amount of offset our method can handle depends on the effective receptive field of the convolution layers in the encoder and decoder networks for image generation, namely E_a, E_p^*, and D. If the offset is too large, the networks may not be able to encode and decode information needed to correctly generate $\tilde{B}_{\theta,\alpha}$. Supporting our intuition, Figure 16 shows that the solution error increases superlinearly with larger initial offset.

Our method, however, naturally allows for a strategy to deal with larger initial offsets at inference time than in the training data, without the need to re-train. At inference, rather than using just B_θ during image generation, we can

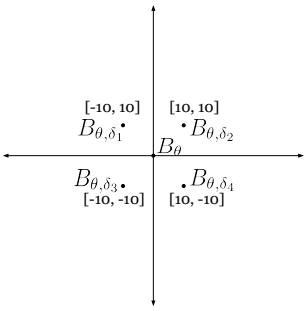


Figure 17. The centre of image B_θ is shifted onto each of the four quadrants to produce four shifted versions.

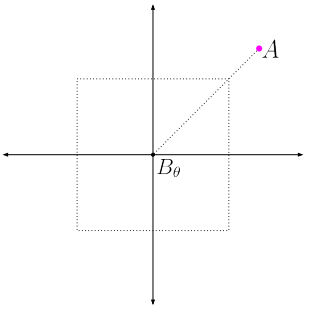


Figure 18. The unknown translation offset α between A and B_θ is larger than the networks are designed for.

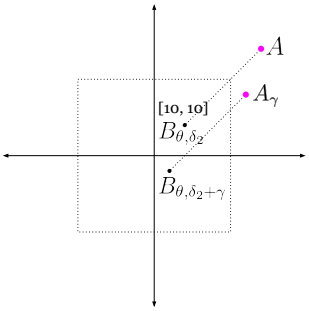


Figure 19. If we shift B_θ by $[10, 10]$ to form B_{θ,δ_2}, then the offset between A and B_{θ,δ_2} is within what the networks are designed for. In this case, generating $\tilde{B}_{\theta,\alpha}$ and $\tilde{B}_{\theta,\alpha+\gamma}$ should both be accurate, as the offset in both cases are within what the networks are trained for.

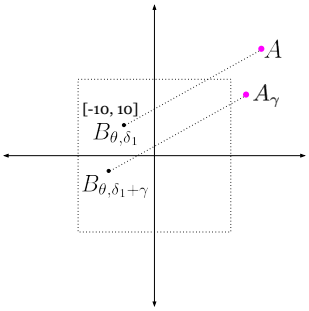


Figure 20. The resulting synthetic image will still be erroneous, if an incorrect quadrant is selected. Here the offset between B_{θ,δ_1} and A is larger than what the networks can handle. In this case, generating $\tilde{B}_{\theta,\alpha}$ and $\tilde{B}_{\theta,\alpha+\gamma}$ will both be problematic due to the issue with offsets.

apply known translation offsets $\delta_1, \delta_2, \delta_3$, and δ_4 to shift the centre of B_θ onto each of the four quadrants. This is depicted in Figure 17, where as an example, we shift B_θ by $[-10, 10]$, $[10, 10]$, $[-10, -10]$, and $[10, -10]$ pixels to form B_{θ,δ_1}, B_{θ,δ_2}, B_{θ,δ_3}, and B_{θ,δ_4}, respectively.

Figure 18 depicts a case where the translation offset α between the satellite image A and B_θ is $\alpha = \begin{bmatrix}35 & 35\end{bmatrix}^T$ pixels. This is larger than the range of translation offsets in the training data, which is $[-25, 25]$ pixels, shown by the dashed box around the origin. However, the offset between B_{θ,δ_2} and A is $\begin{bmatrix}25 & 25\end{bmatrix}^T$ pixels, which is within the range our networks can handle, as shown in Figure 19.

Forming $E_p^*(A, B_\theta)$ and $E_a(B_\theta)$ during image generation might lead to incorrect results when generating $\tilde{B}_{\theta,\alpha}$, as the offset between A and B_θ is larger than what E_p^* is trained for. However, we can also generate $\tilde{B}_{\theta,\alpha}$ using $E_p^*(A, B_{\theta,\delta_2})$ and $E_a(B_{\theta,\delta_2})$:

$$\tilde{B}_{\theta,\alpha} = D\big(E_a(B_{\theta,\delta_2}), E_p^*(A, B_{\theta,\delta_2})\big). \tag{22}$$

It should be noted that such a combination does not suffer from the issue with large offsets, as discussed.

The remaining question is then which shifted image from B_{θ,δ_1}, B_{θ,δ_2}, B_{θ,δ_3}, and B_{θ,δ_4} to choose from. Shown in Figure 20, as an example, generating $\tilde{B}_{\theta,\alpha}$ using $E_p^*(A, B_{\theta,\delta_1})$ and $E_a(B_{\theta,\delta_1})$ will also be problematic, as this combination also suffers from the issue with large offsets. The selection cannot be made ahead of the image generation as α is unknown. However, our method naturally allows for a strategy to select the correct quadrant to shift B_θ onto, without needing to know α a priori, using an introspective method similar in spirit as the one presented in Section 6.4.1.

We can generate five versions of $\tilde{B}_{\theta,\alpha}$ using B_θ, B_{θ,δ_1}, B_{θ,δ_2}, B_{θ,δ_3}, and B_{θ,δ_4}, and introspect the quality of each $\tilde{B}_{\theta,\alpha}$. To do so, we apply a known shift γ to A to query for another image A_γ, and we can also shift each B_{θ,δ_i} by γ to form $B_{\theta,\delta_i+\gamma}$ (or $B_{\theta,\gamma}$ for B_θ), as in Figures 19 and 20.

For each shift δ_i (and zero shift for B_θ), we can take the combination $E_p^*(A_\gamma, B_{\theta,\delta_i+\gamma})$ and $E_a(B_{\theta,\delta_i+\gamma})$ to generate $\tilde{B}_{\theta,\alpha+\gamma}$, which should be pixel-wise aligned with A_γ. If generating $\tilde{B}_{\theta,\alpha}$ is problematic due to large offsets, then so will generating $\tilde{B}_{\theta,\alpha+\gamma}$ be, as shown in Figure 20. On the other hand, if the networks can correctly produce $\tilde{B}_{\theta,\alpha}$, they can also correctly produce $\tilde{B}_{\theta,\alpha+\gamma}$, as shown in Figure 19.

For each shift δ_i, we can compute a translation offset $\hat{\gamma}$ using $\tilde{B}_{\theta,\alpha+\gamma}$ and $\tilde{B}_{\theta,\alpha}$:

$$\hat{\gamma} = \underset{p\in\mathbb{R}^2}{\arg\max}\, \tilde{B}^\dagger_{\theta,\alpha+\gamma} \star \tilde{B}^\dagger_{\theta,\alpha}, \tag{23}$$

along with an error term $e = \|\hat{\gamma} - \gamma\|$. For the five pairs of synthetic images, the one that results in the smallest e will be used and passed downstream to solve for $\hat{\alpha}$. This forms an introspective approach for handling initial offsets larger than what the models are trained for, by augmenting the original input B_θ with various shifted versions.

Table 16 shows results on the RobotCar Dataset for radar localisation against satellite imagery, where the initial offset is now in the range of $[-35, 35]$ pixels. We also include additional results where the networks are trained using the same offset range as at inference for comparison. Taking a model trained for an offset in the range $[-25, 25]$ pixels and evaluating directly with offsets in the range $[-35, 35]$ pixels, we resulted in higher errors compared to Section 6.1. However, with our input-augmentation approach by shifting B_θ and generating $\tilde{B}_{\theta,\alpha}$ multiple times, we can handle larger initial offsets without sacrificing significantly on accuracy. This method, however, introduces additional computational cost as multiple forward passes of image generation are needed.

6.12 Modality Transfer

An interesting application would be to train models using data for localisation between one type of range sensor (e.g., radar) and satellite imagery, and evaluate for localisation between another type of range sensor (e.g., lidar) and satellite imagery. This is particularly useful if we wish to evaluate using a specific type of range sensor at inference, but do not have the associated training data with approximately known trajectories to query for satellite images.

Here we demonstrate how this can be achieved using unsupervised domain adaptation by image-to-image transfer.

Table 16. Radar localisation against satellite imagery evaluated on the test set of RobotCar, where the initial translation offset is large at inference. We also included results from Table 5 for reference.

Offset Range at Training (pixels)	Offset Range at Inference (pixels)	With Input-Augmentation	Mean Error (metric)			Mean Error (pixel)	
			x (m)	y (m)	θ (°)	x	y
$[-25, 25]$	$[-25, 25]$	✗	3.44	5.40	3.03	3.97	6.23
$[-25, 25]$	$[-35, 35]$	✗	6.62	7.88	**3.26**	7.64	9.09
$[-25, 25]$	$[-35, 35]$	✓	4.67	**5.54**	**3.26**	5.39	**6.40**
$[-35, 35]$	$[-35, 35]$	✗	**4.28**	7.35	**3.26**	**4.94**	8.49

Table 17. Localisation results for experiments where the range sensor used is different between training and inference time, with and without using modality transfer, evaluated on the RobotCar Dataset. We also included results from Tables 6 and 10 for reference.

Sensor in Training Data	Sensor at Inference	With Modality-Transfer	Mean Error (metric)			Mean Error (pixel)	
			x (m)	y (m)	θ (°)	x	y
Lidar	Lidar	✗	1.54	1.85	2.29	3.55	4.27
Radar	Lidar	✗	6.39	8.64	4.71	14.74	19.95
Radar	Lidar	✓(Cartesian)	2.63	3.48	2.65	6.08	8.03
Radar	Lidar	✓(Polar)	**2.04**	**2.20**	**2.38**	**4.71**	**5.08**
Radar	Radar	✗	2.23	2.42	3.22	5.29	5.74
Lidar	Radar	✗	7.03	6.43	10.05	16.23	14.83
Lidar	Radar	✓(Cartesian)	2.74	**2.84**	**3.94**	6.32	**6.56**
Lidar	Radar	✓(Polar)	**2.73**	3.52	4.62	**6.31**	8.14

Specifically, we consider CycleGAN (Zhu et al. 2017) for transferring between radar scan images and lidar scan images. For transferring between images of single scans of radar and lidar data, as originally noted in Weston et al. (2020), converting the scan images to polar coordinate representation prior to domain adaptation can lead to more visually realistic results. Figure 15 shows examples of radar and lidar images and their synthetic counterparts generated using our implementation of CycleGAN, for both polar and Cartesian coordinate representations.

At inference, given networks E_a, E_p^*, D, H_B, and $H_{\tilde{B}}$ trained for radar localisation against satellite imagery, and a lidar image B^l, we can generate $\tilde{B}^r = f_{L\to R}(B^l)$, where $\tilde{B}^r$ is a synthetic radar image and $f_{L\to R}$ is a network that transforms a lidar image to a synthetic radar image trained using the objectives of CycleGAN (Zhu et al. 2017).

We can then use $\tilde{B}^r$ as the input for inferring rotation, conditional image generation, and learning pose estimation using the pipeline detailed in Section 4.

The same can be performed for the other way around, where given networks trained for lidar localisation against satellite imagery, we wish to perform radar localisation against satellite imagery at inference time.

Table 17 summarises the results for modality transfer. Networks trained with one type of range sensor suffer from large localisation errors when directly evaluated with another type of sensor. The errors are greatly reduced after applying domain adaptation and using transferred images as input. Shown in Figure 15, applying domain adaptation in polar coordinate representation led to more visually realistic synthetic images when transferring from lidar to radar data, and the smallest localisation errors for networks trained on radar data and tested with lidar data. Applying domain adaptation in Cartesian coordinate representation, on the other hand, resulted in smaller errors for the reverse experiment. We use radar and lidar images of the same resolution in this experiment, and do not consider modality transfers that also involve a change in resolution.

6.13 Further Qualitative Results

Additional qualitative results are presented in Figure 21 showing various stages of our methods for different modalities and datasets.

7 Conclusion and Future Work

We present self-supervised learning to address cross-modality, metric localisation between satellite imagery and on-board range sensors, without metrically accurate ground truth for training. Our approach utilises a multi-stage network that solves for the rotation and translation offsets separately through the generation of synthetic range sensor images as an intermediate step. Our method is validated across a large number of experiments for multiple modes of localisation, with results on-par with a prior supervised approach. A coarse initial pose estimate is needed for our method to compute metric localisation. Therefore, a natural extension would then be to solve place recognition for a range sensor within a large satellite map as a prior step to metric localisation.

Acknowledgements

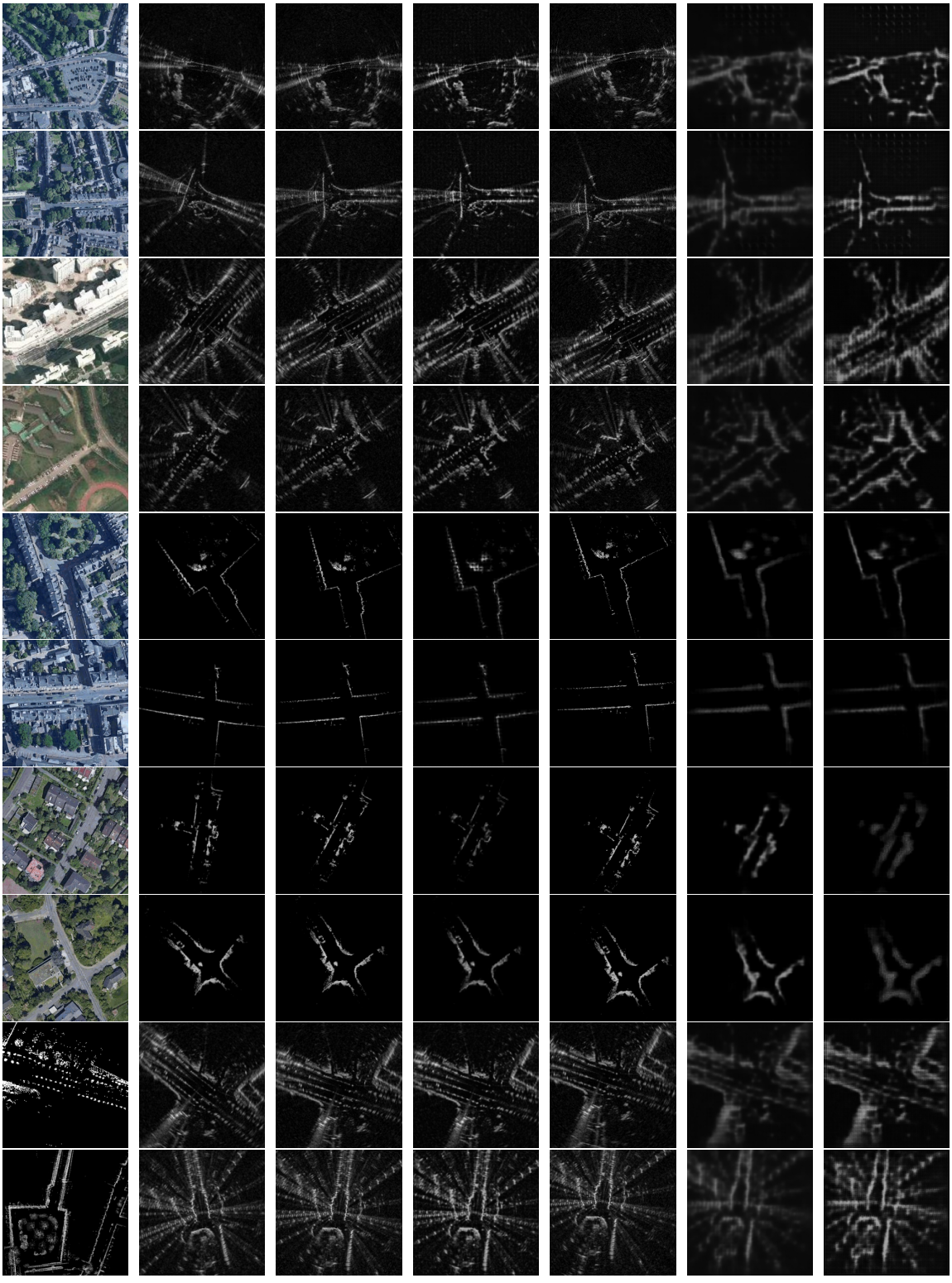
We thank Giseop Kim from IRAP Lab, KAIST for providing GPS data for the MulRan Dataset.

Funding

The work of Tim Y. Tang was jointly supported by a Postgraduate Scholarship - Doctoral Program (PGS-D) from the Natural Sciences and Engineering Research Council of Canada, and an Oxford Robotics Institute Studentship. The work of Daniele De Martini and Paul Newman were supported by EPSRC Programme Grant EP/M019918/1 “Mobile Robotics: Enabling a Pervasive Technology of the Future”. The work of Shangzhe Wu was supported by Facebook Research.

Declaration of conflicting interests

The authors are affiliated with the University of Oxford, UK. Tim Y. Tang was affiliated with the University of Toronto, Canada, until 2018. Daniele De Martini was affiliated with Università degli Studi di Pavia, Italy, until 2018. Shangzhe Wu is concurrently



(a) (b) (c) (d) (e) (f) (g)

Figure 21. Images at various stages of our method: map image A (a), live data image B (b), output of rotation inference B_θ (c), embedding $B^\dagger_\theta$ (d), pixel-wise aligned ground truth $B_{\theta,\alpha}$ (e), synthetic image $\tilde{B}_{\theta,\alpha}$ (f), embedding $\tilde{B}^\dagger_{\theta,\alpha}$ (g). From top to bottom: radar localisation against satellite imagery evaluated on RobotCar (rows 1-2) and MulRan(rows 3-4), lidar localisation against satellite imagery evaluated on RobotCar (rows 5-6) and KITTI (rows 7-8), radar localisation against lidar map evaluated on MulRan (row 9) and RobotCar (row 10).

affiliated with Google Research, and was affiliated with Hong Kong University of Science and Technology, HK, China, until 2018.

References

Aldera R, De Martini D, Gadd M and Newman P (2019) Fast Radar Motion Estimation with a Learnt Focus of Attention Using Weak Supervision. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1190–1196.

Barnes D, Gadd M, Murcutt P, Newman P and Posner I (2020) The Oxford Radar Robotcar dataset: A Radar Extension to

the Oxford Robotcar Dataset. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6433–6438.

Barnes D and Posner I (2020) Under the Radar: Learning to Predict Robust Keypoints for Odometry Estimation and Metric Localisation in Radar. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Paris. URL https://arxiv.org/abs/2001.10789.

Barnes D, Weston R and Posner I (2019) Masking by Moving: Learning Distraction-Free Radar Odometry from Pose Information. In: *Conference on Robot Learning (CoRL)*. URL https://arxiv.org/pdf/1909.03752.

Barsan IA, Wang S, Pokrovsky A and Urtasun R (2018) Learning to Localize Using a LiDAR Intensity Map. In: *CoRL*. pp. 605–616.

Boniardi F, Caselitz T, Kümmerle R and Burgard W (2017) Robust LiDAR-based Localization in Architectural Floor Plans. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 3318–3324.

Broome M, Gadd M, De Martini D and Newman P (2020) On the Road: Route Proposal from Radar Self-Supervised by Fuzzy LiDAR Traversability. *AI* 1(4): 558–585. DOI:10.3390/ai1040033.

Brubaker MA, Geiger A and Urtasun R (2013) Lost! Leveraging the Crowd for Probabilistic Visual Self-localization. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 3057–3064.

Carle PJ and Barfoot TD (2010) Global Rover Localization by Matching Lidar and Orbital 3D Maps. In: *2010 IEEE International Conference on Robotics and Automation*. IEEE, pp. 881–886.

Caselitz T, Steder B, Ruhnke M and Burgard W (2016) Monocular Camera Localization in 3D Lidar Maps. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1926–1931.

Cen SH and Newman P (2019) Radar-Only Ego-Motion Estimation in Difficult Settings via Graph Matching. *arXiv preprint arXiv:1904.11476* .

Chebrolu N, Lottes P, Läbe T and Stachniss C (2019) Robot Localization Based on Aerial Images for Precision Agriculture Tasks in Crop Fields. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1787–1793.

Cho Y, Kim G and Kim A (2019) DeepLO: Geometry-Aware Deep LiDAR Odometry. *arXiv preprint arXiv:1902.10562* .

De Martini D, Gadd M and Newman P (2020) kRadar++: Coarse-to-Fine FMCW Scanning Radar Localisation. *Sensors* 20(21). DOI:10.3390/s20216002.

de Paula Veronese L, de Aguiar E, Nascimento RC, Guivant J, Cheein FAA, De Souza AF and Oliveira-Santos T (2015) 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)*. IEEE, pp. 4285–4290.

Dogruer CU, Koku AB and Dolen M (2010) Outdoor Mapping and Localization Using Satellite Images. *Robotica* 28(7): 1001–1012.

Floros G, Van Der Zander B and Leibe B (2013) OpenStreetSLAM: Global Vehicle Localization Using OpenStreetMaps. In: *2013 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1054–1059.

Geiger A, Lenz P, Stiller C and Urtasun R (2013) Vision Meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)* .

Kaminsky RS, Snavely N, Seitz SM and Szeliski R (2009) Alignment of 3D Point Clouds to Overhead Images. In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, pp. 63–70.

Kim G, Park YS, Cho Y, Jeong J and Kim A (2020) MulRan: Multimodal Range Dataset for Urban Place Recognition. In: *IEEE International Conference on Robotics and Automation (ICRA)*. Submitted. Under Review.

Kingma DP and Ba J (2015) Adam: A method for stochastic optimization. In: Bengio Y and LeCun Y (eds.) *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. URL http://arxiv.org/abs/1412.6980.

Kümmerle R, Steder B, Dornhege C, Kleiner A, Grisetti G and Burgard W (2011) Large Scale Graph-based SLAM using Aerial Images as Prior Information. *Autonomous Robots* 30(1): 25–39.

Lee HY, Tseng HY, Huang JB, Singh M and Yang MH (2018) Diverse Image-to-Image Translation via Disentangled Representations. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. pp. 35–51.

Lenc K and Vedaldi A (2015) Understanding Image Representations by Measuring Their Equivariance and Equivalence. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 991–999.

Leung KYK, Clark CM and Huissoon JP (2008) Localization in Urban Environments by Matching Ground Level Video Images with an Aerial Image. In: *2008 IEEE International Conference on Robotics and Automation*. IEEE, pp. 551–556.

Levinson J and Thrun S (2010) Robust Vehicle Localization in Urban Environments Using Probabilistic Maps. In: *2010 IEEE International Conference on Robotics and Automation*. IEEE, pp. 4372–4378.

Li A, Morariu VI and Davis LS (2014) Planar Structure Matching Under Projective Uncertainty for Geolocation. In: *European Conference on Computer Vision*. Springer, pp. 265–280.

Li Q, Chen S, Wang C, Li X, Wen C, Cheng M and Li J (2019) LO-Net: Deep Real-time Lidar Odometry. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 8473–8482.

Liu MY, Breuel T and Kautz J (2017) Unsupervised Image-to-Image Translation Networks. In: *Advances in Neural Information Processing Systems*. pp. 700–708.

Lu W, Wan G, Zhou Y, Fu X, Yuan P and Song S (2019a) DeepVCP: An End-to-End Deep Neural Network for Point Cloud Registration. In: *The IEEE International Conference on Computer Vision (ICCV)*.

Lu W, Zhou Y, Wan G, Hou S and Song S (2019b) L3-Net: Towards Learning Based LiDAR Localization for Autonomous Driving. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 6389–6398.

Mielle M, Magnusson M and Lilienthal AJ (2019) The Auto-Complete Graph: Merging and Mutual Correction of Sensor and Prior Maps for SLAM. *Robotics* 8(2): 40.

Noda M, Takahashi T, Deguchi D, Ide I, Murase H, Kojima Y and Naito T (2010) Vehicle Ego-Localization by Matching

in-Vehicle Camera Images to an Aerial Image. In: *Asian Conference on Computer Vision*. Springer, pp. 163–173.

Park YS, Jeong J, Shin Y and Kim A (2019) Radar Dataset for Robust Localization and Mapping in Urban Environment . In: *ICRA Workshop on Dataset Generation and Benchmarking of SLAM Algorithms for Robotics and VR/AR*. Montreal.

Parsley MP and Julier SJ (2010) Towards the Exploitation of Prior Information in SLAM. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 2991–2996.

Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J and Chintala S (2019) Pytorch: An imperative style, high-performance deep learning library. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., pp. 8024–8035.

Pink O (2008) Visual Map Matching and Localization Using a Global Feature Map. In: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, pp. 1–7.

Saftescu S, Gadd M, De Martini D, Barnes D and Newman P (2020) Kidnapped Radar: Topological Radar Localisation using Rotationally-Invariant Metric Learning. *arXiv preprint arXiv: 2001.09348* .

Senlet T and Elgammal A (2011) 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)*. IEEE, pp. 2034–2041.

Shu Z, Sahasrabudhe M, Alp Guler R, Samaras D, Paragios N and Kokkinos I (2018) Deforming Autoencoders: Unsupervised Disentangling of Shape and Appearance. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. pp. 650–665.

Tang TY, De Martini D, Wu S and Newman P (2020a) Self-Supervised Localisation between Range Sensors and Overhead Imagery. In: *Robotics: Science and Systems (RSS) XVI*.

Tang TY, Martini DD, Barnes D and Newman P (2020b) RSL-Net: Localising in Satellite Images From a Radar on the Ground. *IEEE Robotics and Automation Letters* 5(2): 1087–1094. DOI: 10.1109/LRA.2020.2965907.

Wang X, Marcotte RJ and Olson E (2019) GLFP: Global Localization from a Floor Plan. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 1627–1632. DOI:10.1109/IROS40897.2019.8968061.

Wang X, Vozar S and Olson E (2017) FLAG: Feature-based Localization between Air and Ground. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3178–3184.

Weston R, Cen S, Newman P and Posner I (2019) Probably Unknown: Deep Inverse Sensor Modelling Radar. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5446–5452.

Weston R, Jones OP and Posner I (2020) There and Back Again: Learning to Simulate Radar Data for Real-World Applications. *arXiv preprint arXiv:2011.14389* .

Wolcott RW and Eustice RM (2014) Visual Localization within Lidar Maps for Automated Urban Driving. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 176–183.

Wolcott RW and Eustice RM (2015) Fast LIDAR Localization Using Multiresolution Gaussian Mixture Maps. In: *2015 IEEE international Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2814–2821.

Wu W, Cao K, Li C, Qian C and Loy CC (2019) Transgaga: Geometry-Aware Unsupervised Image-to-Image Translation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 8012–8021.

Xing X, Han T, Gao R, Zhu SC and Wu YN (2019) Unsupervised Disentangling of Appearance and Geometry by Deformable Generator Network. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 10354–10363.

Xu Y, John V, Mita S, Tehrani H, Ishimaru K and Nishino S (2017) 3D Point Cloud Map Based Vehicle Localization Using Stereo Camera. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 487–492.

Zhu JY, Park T, Isola P and Efros AA (2017) Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In: *Proceedings of the IEEE International Conference on Computer Vision*. pp. 2223–2232.

5

Point-based Metric and Topological Localisation between Lidar and Overhead Imagery

This work was submitted as a journal article to Autonomous Robots, and has been fully accepted in January 2023 for publication. A conference version of this article, "Get to the Point: Learning Lidar Place Recognition and Metric Localisation Using Overhead Imagery" was published and presented at Robotics: Science and System XVII in July 2021, and included in appendix C for completeness.

Background and Motivation

In prior chapters, we targeted only the image registration, or metric localisation between range sensor data and overhead imagery. For outdoor autonomous navigation, it is often desirable to have a standalone system capable of solving both place recognition and metric localisation. In this chapter, we present a novel learned method that solves both using overhead imagery only. Different from the work in Chapters 3 and 4, which seeks to generate synthetic range sensor images from satellite imagery, here we address the modality difference by representing both forms of data as 2D point sets.

Point-based Metric and Topological Localisation between Lidar and Overhead Imagery

Tim Yuqing Tang[1*], Daniele De Martini[1] and Paul Newman[1]

[1]Mobile Robotics Group, Oxford Robotics Institute, University of Oxford, UK.

*Corresponding author(s). E-mail(s): ttang@robots.ox.ac.uk;
Contributing authors: daniele@robots.ox.ac.uk; pnewman@robots.ox.ac.uk;

Abstract

In this paper, we present a method for solving the localisation of a ground lidar using overhead imagery only. Public overhead imagery such as Google satellite images are readily available resources. They can be used as the map proxy for robot localisation, relaxing the requirement for a prior traversal for mapping as in traditional approaches. While prior approaches have focused on the metric localisation between range sensors and overhead imagery, our method is the first to learn both place recognition and metric localisation of a ground lidar using overhead imagery, and also outperforms prior methods on metric localisation with large initial pose offsets. To bridge the drastic domain gap between lidar data and overhead imagery, our method learns to transform an overhead image into a collection of 2D points, emulating the resulting point-cloud scanned by a lidar sensor situated near the centre of the overhead image. After both modalities are expressed as point sets, point-based machine learning methods for localisation are applied.

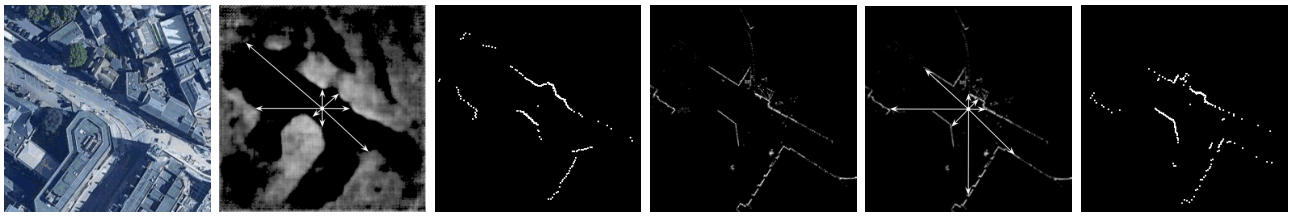
Keywords: Localisation, cross-modality localisation, deep learning, learning from points

1 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 decades. We distinguish 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 outdoor vehicle localisation due to its sensing range and invariance to lighting conditions. Both place recognition [1–4] and metric localisation [5–8] using lidar are well-studied problems, as well as methods that target global localisation [9–12] 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. This paper presents an alternative approach to localise a lidar using only off-the-shelf, easily accessible overhead imagery such as Google satellite imagery, which often captures geometric entities also observable by ground lidar scans, providing cues for localisation.

The significant modality difference between overhead imagery and ground range sensors is an



(a) (b) (c) (d) (e) (f)

Fig. 1: Left ((a) to (c)): given an overhead image (a), we learn an occupancy image (b) that indicates how likely each pixel will produce a point return if scanned by a lidar. This results empirically in a 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 (c). Right ((d) to (f)): a lidar scan taken nearby, rotation-aligned with the overhead image for better visualisation (d). Ray-tracing the lidar image and keeping only the first return for each azimuth (e), we obtain a point-cloud (f) that can be directly compared against the pseudo point-cloud (c).

immediate challenge. The work in [13] tackles the modality difference between satellite imagery and radar by synthetising a radar image from a pair of proximal satellite and radar images. The synthetic image is then in the same domain as the live sensor and can be used for registration. The experimental results using the method in [13] are demonstrated for lidar metric localisation against overhead imagery in the authors' follow-up work [14, 15].

The method in [13] has two significant limitations. Firstly, it requires geometrically proximal (but not necessarily pose-aligned) pairs of satellite and radar images to address the modality difference for metric localisation. Therefore, it requires place recognition to have been previously solved by an external system. Secondly, the quality of the image generation in [13] is extremely dependent on the initial offset, resulting in pose errors increasing disproportionally with larger initial heading offsets. While the methods in [13–15] address the modality difference by generating synthetic range sensor images from overhead imagery, we take a different approach where both overhead images and lidar scans are converted into a common representation, 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 first learn an occupancy image that indicates how likely each pixel is to induce a range return if scanned 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 sidewalks. We then ray-trace from near the centroid of the occupancy image along each azimuth until the first "occupied" pixel, which emulates the resulting point-cloud as if a lidar near the centroid takes a scan of its surrounding environment. This process is depicted in Figure 1 together with an actual lidar scan taken at roughly the same position for visual reference.

To the best of the authors' knowledge, our proposed method is the first to learn the place recognition of a ground range sensor using only publicly available overhead imagery, with no GPS or prior sensor maps at inference time. We evaluate our method for both place recognition and metric localisation on publicly available datasets, showing that we outperform the prior work in [13, 14] 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 after applying appropriate domain transfer on the input sensor data.

This article is an extended version of our prior work [16]. The improvements include a more thorough explanation of our motivation for using a point-based approach in contrast with prior works that are image-based [13–15] in Section 3.1, and a more in-depth description of our method in Section 4. In terms of experimental validation (Section 5), we include additional qualitative results and an ablation study on using reduced training data. We also show how registration errors can be consistently reduced using an improved network architecture, and present a new loss function for learning place recognition that leads to higher retrieval accuracy. We further demonstrate that simply utilising a sequential data stream rather than a single frame of data for place recognition at test time significantly improves the retrieval accuracy without needing to retrain the network. Finally, we include additional

results on test data across different sequences than in the training data to show the capability of our method to not only generalise to unseen places, but also to sensor data recorded over different days.

2 Related Work

2.1 Range Sensor Localisation against Overhead Imagery

While many works were proposed for visual localisation using overhead imagery [17–22], localising a ground range sensor remains a less-targeted problem due to the challenging modality difference.

Similar to dense image registration, [23] 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 [24, 25] pre-process the overhead images by performing edge detection and semantic segmentation respectively, before comparing against range sensor data. Hussein et al. [26] 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 [27] localised a robot equipped with a lidar within orbital elevation maps by detecting features from topographic peaks. These methods are examples of hand-crafted methodologies designed for specific setups and may not necessarily generalise to more complex scenarios.

In light of this, RSL-Net [13] 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. The approach in [13] was later extended for self-supervised learning in the authors' follow-up work [14, 15]. However, the methods in [13–15] remain strictly limited to metric localisation, and cannot solve place recognition. Concurrent work by Zhu et al. [28] propose to learn a correction and matching probability between lidar grid-map and satellite imagery, which are used as factors in a pose graph optimisation to correct for drift in lidar odometry, but do not directly solve the metric localisation problem alone. Recent work in [29] learns to solve the registration between ground camera and lidar data and overhead imagery by maximising the correlation in feature space over discrete poses, similar as in [13], but require ground camera data in addition to range sensor scans.

Different than prior work in [13–15, 28, 29], our proposed method does not require ground camera data, and can handle both the metric localisation and place recognition between a ground range sensor and overhead imagery.

2.2 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 [30, 31] localise by matching odometry trajectories to road paths from OSM. Panphattarasap and Calway [32] 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. [33] utilised a similar strategy for localisation using OSM but relied only on existing networks trained for point-cloud semantic segmentation, eliminating the need to train new networks for recognising intersections and gaps.

Our method differs from [30, 31] in that we do not need a sequence of on-board measurements to compute odometry before a solution can be found. Compared to [32, 33], 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.

2.3 Learning-based Point-Cloud Registration

After the overhead image is expressed as 2D points, we seek to solve for its registration against points from a lidar scan. Learning-based methods for point-cloud registration have shown to be less prone to converging to local minima than classical approaches such as ICP [34]. The method in [35] learns local RGB-D descriptors with 3D convolutions for matching. 3DFeatNet [8] samples point-clusters using PointNet++ [36] and learns local descriptors using a triplet loss on point-clusters for feature alignment. Given a set of feature descriptors, the method in [37] learns weights for point correspondences for downstream

pose estimation, acting similarly in spirit as an outlier-rejection step. DCP [38] learns a per-point descriptor using DGCNN [39], finds point correspondences by matching learned descriptors, and computes the pose offset using singular value decomposition (SVD). In particular, the descriptors in [38] 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 [40] builds upon DCP, allowing for partial-to-partial registration, by using Gumbel-Softmax for matching and iterative alignment for registration. DeepGMR [41] learns point-to-distribution rather than point-to-point correspondences by representing a point-cloud as a Gaussian Mixture during descriptor learning, greatly reducing complexity for large point numbers.

We utilise DCP [38] for registration and learning descriptors, as it outperforms other methods for our problem in our experiments.

2.4 Deep Learning for Point-Cloud Place Recognition

Several learning-based methods were proposed for large-scale point-cloud place recognition. PointNetVLAD [42] combines PointNet [43] and NetVLAD [44] to learn a per-point-cloud global descriptor for retrieval. LPD-Net [45] aggregates learned local features to produce a global descriptor for place recognition, outperforming [42]. PCAN [46] utilises attention to predict each local point feature's significance and favours task-relevant features when aggregating local features into a global one for retrieval. DH3D [47] 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 learning a global descriptor for place recognition. Barnes and Posner [48] learned key-points from radar images for odometry, and pooled local descriptors across spatial dimensions into a global one per image for place recognition. Recently, OverlapNet [49] was proposed to detect loop closures based on the overlap between lidar images.

Our approach to learning place recognition is similar to [47] 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 [38] with a loss on the pose. Re-using local descriptors learned for registration ensures the global descriptor is based on geometrically distinctive features, which we know provide valuable clues for place recognition.

3 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 $x-y$ plane from a top-down perspective, producing 2D, "birds-eye" view lidar images where pixels with a non-zero intensity indicate range returns. We discard points with z values less than 0 when creating the lidar images to remove returns on the ground. We denote satellite images and lidar images as I^S and I^L, respectively. Furthermore, we pre-process all overhead and lidar images to have the same resolution (or scale factor), so any registration between lidar data and overhead imagery is in $SE(2)$.

For place recognition, we consider a set of satellite images queried along known routes within an operating environment, $\{\mathrm{I}^S_i\}, i = 1, 2, \cdots, N_s$, and lidar images taken across time-stamps during live traversal $\{\mathrm{I}^L_j\}, j = 1, 2, \cdots, N_L$. For each I^L_j, we seek to find its nearest spatial neighbour from $\{\mathrm{I}^S_i\}$; this provides a coarse estimate of the vehicle's global position as the latitude and longitude coordinates of each I^S_i 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^L_j, there is at least one correct spatial match from the set $\{\mathrm{I}^S_i\}$, and vice versa. Though this is not a hard requirement for our algorithm at inference time (as described in detail in Section 4.5), since each I^L_j is compared against all satellite images $\{\mathrm{I}^S_i\}$ to find the best match, regardless of the size N_S. During training, however, we required pose-aligned pairs of lidar and satellite images to form supervision, as detailed in Section 4.1, and therefore N_L and N_S must be the same.

For metric localisation, instead, given the spatially proximal images $\mathrm{I}^S_i \in \{\mathrm{I}^S_i\}$ and $\mathrm{I}^L_j \in \{\mathrm{I}^L_j\}$

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

we seek to solve their unknown $SE(2)$ initial pose offset expressed as $\begin{bmatrix} x & y & \theta \end{bmatrix}$. For simplicity, the indices i and j will be dropped in the context of metric localisation.

Though in practise, typically a coarse pose estimate is found by place recognition first and then refined by metric localisation, our approach requires learning the metric localisation first prior to learning place recognition.

3.1 Image Generation vs. Point Learning

A widely used strategy for dealing with modality differences between two types of images is to apply image-to-image transfer methods such as Pix2Pix [50] or CycleGAN [51]. In a naive image-to-image transfer approach, a neural network approximates the following mapping:

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

where $\hat{\mathrm{I}}^L$ is a synthetic lidar image pose-aligned with I^S. However, standard image-to-image transfer methods are not appropriate for transferring between satellite and lidar images. Due to occlusion and a range sensor's limited field-of-view (FOV) compared to overhead images, the resulting lidar scan depends on not just the surrounding environment, but also the centroid at which the scan is taken. Because of this, in some cases, the same patch in satellite imagery may contain lidar returns in one satellite-lidar pair, but none in another, as shown in Figure 2. Therefore, within the training data, the mapping g as in Equation (1) is one-to-many on the level of image patches. Since a one-to-many mapping is not a function, the mapping g in (1) is not suitable for learning with neural networks, which are function approximators.

To handle such ambiguity, the image generation in RSL-Net [13] is conditioned on both a satellite image and a live radar image, where the latter dictates the exact appearance of the synthetic image, which in turn is pixel-wise aligned to the satellite image. As CNNs are non-equivariant[2] to rotation [52], RSL-Net seeks to infer the rotation offset prior to image generation:



Fig. 2: Two satellite-lidar image pairs taken 10 seconds apart, where lidar measurements (white points) are overlaid on top of satellite images with ground truth pose. The resulting lidar scan depends not just on the surrounding scene, but also on the centroid where it is taken. Shown here, 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.

$$h(\mathrm{I}^L, \mathrm{I}^S) \rightarrow \mathrm{I}^L_*, \quad g(\mathrm{I}^L_*, \mathrm{I}^S) \rightarrow \hat{\mathrm{I}}^L, \tag{2}$$

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, respectively. In particular, the mapping g in (2) is now a one-to-one mapping [15] and therefore can be properly approximated by a neural network. However, since the input to h and g in Equation (2) are paired satellite and lidar data that are spatially proximal, RSL-Net requires an existing place recognition solution and cannot be used to solve place recognition on its own. This is also true in the self-supervised image-based approaches in [14, 15].

Furthermore, it can be immediately seen that because CNNs are not equivariant to rotation, the performance of image generation in network g of Equation (2) 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 the generated images.

We show in Section 4 that unlike the mapping g in Equation (1) which attempts to transform a satellite image directly into a lidar image, our method learns an occupancy image first before

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

transforming to a domain that is comparable to lidar data. This two-step approach is inherently occlusion aware, does not introduce ambiguity as in Equation (1), and requires no paired input data at inference time.

4 Methodology

Our proposed pipeline follows a number of steps. First, given an overhead image, we propose a novel method to represent it as a collection of 2D points. We start by learning an occupancy image using lidar data as the supervision signal, as detailed in Section 4.1. Points are then extracted from the occupancy image with ray-tracing, forming a pseudo point-cloud, as described in Sections 4.2 and 4.3. Next, we train a DCP [38] network to solve the registration between the pseudo point-cloud and points extracted from an actual lidar image, as in Section 4.4. Finally, we take the local per-point descriptors from the DCP network optimised for registration, and re-use them to learn a per-point-cloud global descriptor for place recognition with PointNetVLAD [42], as detailed in Section 4.5.

4.1 Learning Lidar Occupancy from Overhead Imagery

Given a satellite image I^S, we learn an occupancy image $\hat{\mathrm{O}}$, which provides information on the likelihood of each pixel to cause a range return if a lidar situated near the image centroid takes a scan. Suppose ground truth occupancy label O exists, and $\mathrm{O}(i,j) \in \{0,1\}$, where $\mathrm{O}(i,j) = 1$ indicates pixel $[i,j]$ belongs to "occupied" space, such as on buildings, and $\mathrm{O}(i,j) = 0$ indicates "free-space", such as on streets and side-walks. We wish to compute the probability map $P(\mathrm{O}|\mathrm{I}^s, \Theta)$ modelled by Bernoulli distributions over all pixels and approximated by a neural network f_o with parameters Θ :

$$f_o(\mathrm{I}^S) \rightarrow \hat{\mathrm{O}}. \tag{3}$$

To optimise the network f_o, we minimise the weighted binary cross entropy (BCE) loss:

$$\begin{aligned} \mathcal{L}_{\mathrm{BCE}} &= -\log P\big(\mathrm{O}|\mathrm{I}^s;\ \Theta\big) \\ &= -\sum_{i,j} \log P\big(\mathrm{O}(i,j)|\mathrm{I}^s;\ \Theta\big) \\ &= -\sum_{i,j} \omega_{ij} \big[\mathrm{O}(i,j) \cdot \log \hat{\mathrm{O}}(i,j) + \\ &\quad \big(1 - \mathrm{O}(i,j)\big) \cdot \log\big(1 - \hat{\mathrm{O}}(i,j)\big)\big], \end{aligned} \tag{4}$$

where ω_{ij} is the scalar weight for pixel $[i,j]$.

In practice, the pixel values in $\hat{\mathrm{O}}$ are in the range $\big[0,1\big]$, where larger values indicate a higher probability that a pixel is on occupied space, and smaller values indicate free-space.

The training of $\hat{\mathrm{O}}$ as in Equation (4) is realised by finding the appropriate ground truth occupancy label O and weights ω_{ij}, which we can obtain by taking into account the sensing nature of lidars. As also noted in [53], in a lidar image with no ground points, by ray-tracing from the centroid along each azimuth, pixels from the centroid to immediately in front of the first range return are 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 first create 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. We can then construct a binary certainty mask M from $\mathrm{I}^{L\dagger}$ by ray-tracing again along each azimuth, as shown in Figure 4, where $\mathrm{M}(i,j) = 1$ indicates we are certain about whether pixel $[i,j]$ is occupied or free, and $\mathrm{M}(i,j) = 0$ indicates we are uncertain. This process can be performed efficiently in parallel by transforming the lidar image to polar (range-azimuth) coordinate representation with a finite number of discretised azimuths. In our experiments, we set the threshold γ to 0.2.

We can then use $\mathrm{I}^{L\dagger}$ as the ground truth label and M as the weights for training f_o. Substituting into Equation (4), the loss for learning the

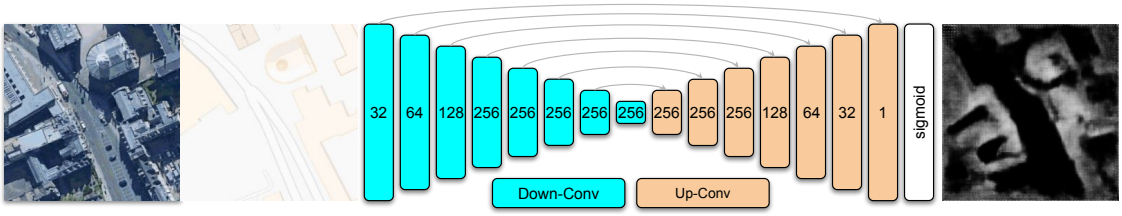


Fig. 3: To generate the occupancy image $\hat{\mathrm{O}}$, we use a U-Net architecture with 8 down-sample convolution blocks (each followed by Leaky ReLU and BatchNorm) and 8 up-sample convolution blocks (each followed by ReLU, BatchNorm, and dropout, except the last block), with skip connections between layers. We use 4×4 kernels with a stride of 2. 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.

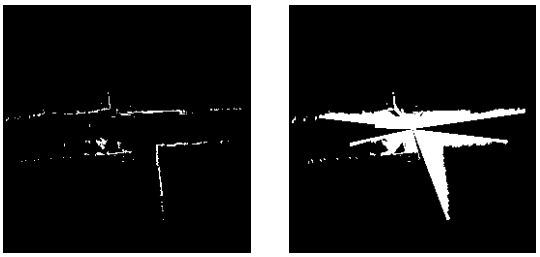


Fig. 4: Given a lidar image I^L we first threshold I^L to produce a binary lidar image $\mathrm{I}^{L\dagger}$ (left) where $\mathrm{I}^{L\dagger}(i,j)$ is 0 or 1 for any pixel $[i,j]$, with 1 indicating return and 0 indicating no return. Ray-tracing $\mathrm{I}^{L\dagger}$, pixels from the centroid 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.

occupancy image becomes

$$\mathcal{L}_{\text{occ}} = -\sum_{i,j} \mathrm{M}(i,j)\big[\mathrm{I}^{L\dagger}(i,j) \cdot \log \hat{\mathrm{O}}(i,j) + \big(1 - \mathrm{I}^{L\dagger}(i,j)\big) \cdot \log\big(1 - \hat{\mathrm{O}}(i,j)\big)\big]. \tag{5}$$

Notably, the loss only back-propagates through a small fraction of pixels where we are certain about the occupancy information ($\mathrm{M}(i,j) = 1$), rather than through all pixels when using an L_1 loss for image generation as in RSL-Net [13]. 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, preventing the ambiguity in learning a one-to-many mapping as described in Section 3.1.

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

$$f_o(\mathrm{I}^S, \mathrm{I}^R) \rightarrow \hat{\mathrm{O}}, \tag{6}$$

where I^R is a roadmap image queried at the same location as I^S, and the probability map we wish to compute becomes $P(\mathrm{O}|\mathrm{I}^S, \mathrm{I}^R, \Theta)$. We parametrise f_o using a U-Net architecture [54], shown in Figure 3. We also experimented with other common architectures for image-to-image transfer such as variations of ResNet-based [55] generators, but the U-Net architecture as in Figure 3 resulted in the best performance, and was particularly useful for preserving the location of building edges. We show in Figure 5 that, at inference time when evaluated on unseen images, the auxiliary input I^R bolsters the robustness of f_o against regions in I^S with a strong pixel intensity gradient but not corresponding to a boundary between free and occupied space, such as shadows. We pretrain f_o for 100 epochs with random rotations on the input images for data augmentation, prior to downstream tasks.

4.2 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. Similar 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

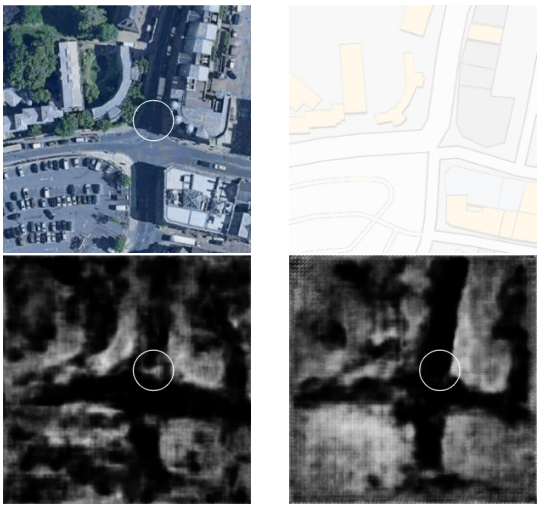


Fig. 5: Top left: an unseen satellite image I^S at inference. Top right: the corresponding I^R queried at the same location as I^S. The circled region corresponds to a strong gradient in I^S caused by shadows, and results in some pixels in this region inferred as occupied where they should be on free-space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

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 set both N_A and N_R, the number of discretised azimuths and ranges, respectively, to 256. The result is the point-cloud $\mathbf{P}^S \in \mathbb{R}^{256\times 2}$. A secondary result of Algorithm 1 is $\mathbf{s} \in \{0,1\}^{N_A}$, a binary score which is set to 1 if a specific azimuth contains a return and 0 otherwise; this score will be used later in the pipeline (see Equation (9)).

4.3 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 with radius $\frac{1}{2}$ for $\mathbf{P}^S$.

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, the 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

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
γ // 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 // iterate to the next i
    end if
  end for
end for

$[i^*, j^*]$, 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 ray-tracing process itself in Algorithm 1, can be conducted efficiently in parallel with tensor operations.

4.4 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 [38] to solve

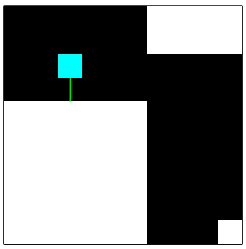


Fig. 6: A 10×10 image patch where the centroid is on occupied space, which is labelled as white pixels. For a particular pixel in this patch (cyan), we can compute its distance (green) to the nearest occupied pixel. The pixel within the patch that is the furthest away from its nearest occupied pixel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

for their pose difference. DCP was shown to outperform ICP on non-identical point sets [38] and, in addition, learns descriptors that can be used later for place recognition. DCP utilises DGCNN [39] and a Transformer [56] module to compute a d-dimensional descriptor per point, performs a soft matching and uses SVD for pose estimation. As detailed in Section 4.4.1, when computing the covariance matrix in SVD, we weight each correspondence using the score vector $\mathbf{s}$ from Algorithm 1, such that azimuths in $\hat{\mathrm{O}}$ with no returns do not contribute to pose estimation.

DCP outputs an estimated rotation matrix $\hat{\mathbf{R}} \in \mathbb{R}^{2\times2}$ and a translation vector $\hat{\mathbf{t}} \in \mathbb{R}^2$. We can establish a loss term using ground truth $\mathbf{R}$ and $\mathbf{t}$:

$$\mathcal{L}_{\text{pose}} = \left\|\hat{\mathbf{t}} - \mathbf{t}\right\|_1 + \lambda \left\|\hat{\mathbf{R}}\mathbf{R}^T - \mathbf{I}\right\|_1, \qquad (7)$$

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 [38], 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 (7) to not only optimise the DCP network, but also fine-tune the pretrained network for f_o, without needing to further apply the loss in Equation (5). The data flow for pose estimation is shown on the left of Figure 7.

4.4.1 SVD for Pose Estimation

Here we give details on the SVD algorithm in our implementation of DCP. Specifically, our implementation differs from the original implementation in [38] in that the point-cloud registration is in $SE(2)$ rather than $SE(3)$, and that we weight the correspondences using a score vector $\mathbf{s}$. Further details on using SVD for point-cloud registration can be found in [57].

Given two proximal point-clouds $\mathbf{P}^S$ and $\mathbf{P}^L$, the DGCNN+Transformer module in DCP computes soft correspondences $\mathbf{P}^{L'}$, for each $\mathbf{p}_i^S \in \mathbf{P}^S$. Since DCP uses soft matching, each $\mathbf{p}_i^{L'} \in \mathbf{P}^{L'}$ is a weighted softmax of all point coordinates in $\mathbf{P}^L$.

We begin by finding the centroids:

$$\bar{\mathbf{p}}^S = \frac{1}{N}\sum_i \mathbf{p}_i^S, \qquad \bar{\mathbf{p}}^{L'} = \frac{1}{N}\sum_i \mathbf{p}_i^{L'}, \qquad (8)$$

where each $\mathbf{p}_i^S$ and $\mathbf{p}_i^{L'}$ is a point in $\mathbf{P}^S$ and $\mathbf{P}^{L'}$, respectively.

The covariance matrix for SVD is then:

$$\mathbf{H} = \sum_i (\mathbf{p}_i^S - \bar{\mathbf{p}}^S) s_i (\mathbf{p}_i^{L'} - \bar{\mathbf{p}}^{L'})^T, \qquad (9)$$

where $\mathbf{H} \in \mathbb{R}^{2\times2}$, and $s_i \in \mathbf{s}$ is the score for $\mathbf{p}_i^S$ from Algorithm 1. We can then apply singular value decomposition on the covariance:

$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \text{SVD}(\mathbf{H}). \qquad (10)$$

The estimated rotation is found by:

$$\hat{\mathbf{R}} = \mathbf{V}\begin{bmatrix}1 & 0\\ 0 & \det(\mathbf{V}\mathbf{U}^T)\end{bmatrix}\mathbf{U}^T, \qquad (11)$$

and finally the estimated translation is found by:

$$\hat{\mathbf{t}} = \bar{\mathbf{p}}^{\mathbf{L}'} - \hat{\mathbf{R}}\bar{\mathbf{p}}^{\mathbf{S}}, \qquad (12)$$

where $\hat{\mathbf{R}} \in \mathbb{R}^{2\times2}$ and $\hat{\mathbf{t}} \in \mathbb{R}^2$.

4.5 Learning Place Recognition

Shown in Figure 7, as a middle step, the DGCNN+Transformer module in DCP outputs local descriptors $\mathbf{D}^S$ and $\mathbf{D}^L$ taking the point coordinates as input. After the networks are

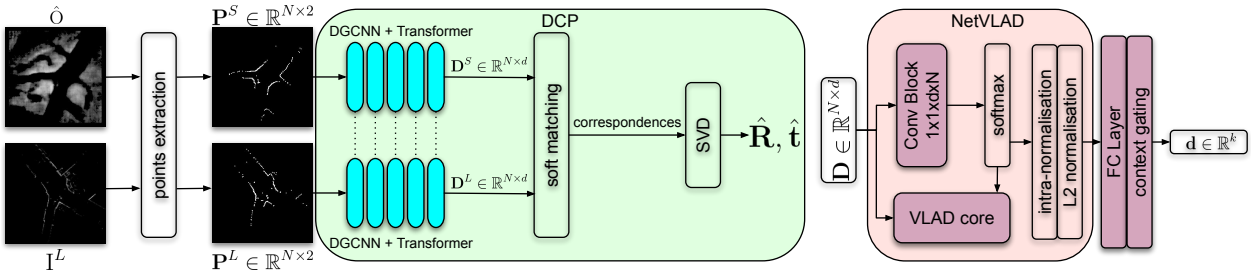


Fig. 7: Data flow at forward pass. Left: for pose estimation, given a spatially proximal pair of $\hat{\mathrm{O}}$ and I^L, we extract point-clouds $\mathbf{P}^S$ and $\mathbf{P}^L$ respectively, using Algorithm 1. The DGCNN+Transformer module in DCP learns for each point-cloud local descriptors, then 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 rotation-invariant global descriptor $\mathbf{d} \in \mathbb{R}^k$, which can be used for retrieval.

trained for pose estimation, the local descriptors are optimised to store geometric information useful for registration.

Given a point-cloud $\mathbf{P} \in \mathbb{R}^{N\times 2}$ and its local descriptors $\mathbf{D} \in \mathbb{R}^{N\times d}$ ($\mathbf{D}$ can be either $\mathbf{D}^S$ or $\mathbf{D}^L$ depending on whether $\mathbf{P}$ is $\mathbf{P}^S$ or $\mathbf{P}^L$), we use a PointNetVLAD [42] 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 loss to encourage descriptors from positive (spatially proximal) pairs of $\mathbf{P}^S$ and $\mathbf{P}^L$ to have a smaller L_2 distance than negative (spatially distant) pairs. The global descriptor is made rotation-invariant by perturbing the point-clouds with arbitrary random rotations during training. In other words, while positive pairs of $\mathbf{P}^S$ and $\mathbf{P}^L$ have proximal centre positions, they can have any arbitrary heading offset. In our prior work [16], a standard triplet loss is used where an anchor point-cloud from lidar, $\mathbf{P}^L$, is used to form positive and negative pairs with pseudo point-clouds $\mathbf{P}^{S+}$ and $\mathbf{P}^{S-}$. Formally, this is

$$\mathcal{L}_{\text{pr}} = \Big[\left\|\mathbf{d}^{S+} - \mathbf{d}^{L}\right\|_2 - \left\|\mathbf{d}^{S-} - \mathbf{d}^{L}\right\|_2 + m \Big]_+, \tag{13}$$

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 $\mathbf{P}^L$ from a lidar image I^L. $\mathbf{d}^{S+}$ is the global descriptor for $\mathbf{P}^{S+}$ obtained 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.

While the standard triplet loss as in Equation (13) is widely used for descriptor learning in place recognition, several improvements to the triplet loss were introduced recently, such as the quadruplet loss [58]. Building upon the loss in Equation (13), we propose a bidirectional triplet loss:

$$\begin{aligned}\mathcal{L}_{\text{pr}} = &\Big[\left\|\mathbf{d}^{S+} - \mathbf{d}^{L}\right\|_2 - \left\|\mathbf{d}^{S-} - \mathbf{d}^{L}\right\|_2 + m \Big]_+ + \\ &\Big[\left\|\mathbf{d}^{L+} - \mathbf{d}^{S}\right\|_2 - \left\|\mathbf{d}^{L-} - \mathbf{d}^{S}\right\|_2 + m \Big]_+ .\end{aligned} \tag{14}$$

In addition to the anchor lidar image I^L, we also randomly sample an anchor satellite and roadmap image pair $\mathrm{I}^S, \mathrm{I}^R$, from the training set to form an anchor pseudo point-cloud $\mathbf{P}^S$ and its associated global descriptor $\mathbf{d}^S$. $\mathbf{d}^{L+}$ and $\mathbf{d}^{L-}$ are the global descriptors for $\mathbf{P}^{L+}$ and $\mathbf{P}^{L-}$, which form positive and negative pairs with $\mathbf{P}^S$, respectively. The idea to the bidirectional triplet loss in Equation (14) is simple: if given a lidar point-cloud $\mathbf{P}^L$ the network can distinguish between proximal and distant samples of $\mathbf{P}^S$, then it should also be able to do the reverse. The loss in Equation (14) further encourages the PointNetVLAD layer to learn globally distinctive descriptors, and leads to consistently better retrieval accuracy than the loss in Equation (13), as shown in Section 5.2.

4.6 Implementation Details

Here we provide various implementations details on the network architectures, design parameters, and our training strategy.

4.6.1 Network Design Choices

For learning the occupancy images, we experimented with a number of network architectures for parametrizing f_o, and we found that a simple and well-established U-Net outperformed other architectures at preserving the locations of building edges. For learning point-cloud registration, we experimented with DCP [38], PRNet [40], and DeepGMR [41], and found that DCP led to the smallest registration errors. In particular, although PRNet was shown to outperform DCP in the authors' experiments, using Softmax (as in DCP) is more appropriate than Gumbel-Softmax (as in PRNet) in our problem, since $\mathbf{P}^S$ and $\mathbf{P}^L$ are from different data sources, and there is often no perfect one-to-one point correspondence when performing point matching. Finally, we use PointNetVLAD for place recognition as it is an intuitive choice for re-using optimised local descriptors to construct a global descriptor, similar as done in prior work [47].

For DCP we implemented using the official open source code[3] with the default architectures for DGCNN and Transformer, but changed the SVD algorithm from solving in $SE(3)$ to $SE(2)$, which we described in detail in Section 4.4.1. Here, we set the descriptor length to 1024 and the number of heads in the Transformer module to 16. For PointNetVLAD, we followed the authors' official open source implementation[4], and set the cluster size to 8, the maximum number of samples to 256, and the output dimension to 2056. We outline the detailed architectures for our implementation of U-Net in Table 1, where we make use of the following abbreviations:

- Conv(C_i, C_o, ks, str): 2D convolution with input channel C_i, output channel C_o, kernel size ks, stride str, and zero padding
- BN: batch normalization
- ReLU: rectified linear unit
- LReLU(s): leaky ReLU with negative slope s
- Drop(d): drop out with rate d
- ConvT(C_i, C_o, ks, str): transposed 2D convolution with input channel C_i, output channel C_o, kernel size ks, stride str, and zero padding

[3] https://github.com/WangYueFt/dcp
[4] https://github.com/mikacuy/pointnetvlad

Table 1: Architecture of our implementation of U-Net for learning occupancy images

Input shape - $6 \times 256 \times 256$ where $C = 6$, $H = W = 256$
Conv(6, 32, 4, 2) + BN + LReLU(0.2)
Conv(32, 64, 4, 2) + BN + LReLU(0.2)
Conv(64, 128, 4, 2) + BN + LReLU(0.2)
Conv(128, 256, 4, 2) + BN + LReLU(0.2)
Conv(256, 256, 4, 2) + BN + LReLU(0.2)
Conv(256, 256, 4, 2) + BN + LReLU(0.2)
Conv(256 256, 4, 2) + BN + LReLU(0.2)
Conv(256, 256, 4, 2) + BN + LReLU(0.2)
ConvT(256, 256, 4, 2) + BN + ReLU + Drop(0.8)
ConvT(256, 256, 4, 2) + BN + ReLU + Drop(0.8)
ConvT(256, 256, 4, 2) + BN + ReLU + Drop(0.8)
ConvT(256, 128, 4, 2) + BN + ReLU + Drop(0.8)
ConvT(128, 64, 4, 2) + BN + ReLU + Drop(0.8)
ConvT(64, 32, 4, 2) + BN + ReLU + Drop(0.8)
ConvT(32, 1, 4, 2) + BN + ReLU + Sigmoid()
With skip connections between connected layers as shown in Figure 3
Output shape - $1 \times 256 \times 256$

4.6.2 Parameter Choices

An important parameter is the patch size h to search around the centroid when finding the origin for ray-tracing, as detailed in Section 4.3. Intuitively, h should be large enough in order for an adequate ray-tracing origin to be found. However, choosing a search space that is too large may introduce additional computational burden, and risk setting the origin to be too far away from the image centroid. Empirically, we set h to 24 as it was sufficiently large for a decent origin to always be found.

The threshold γ for creating the binary lidar image $\mathrm{I}^{L\dagger}$ in Section 4.1 and forming the pseudo point-cloud in Algorithm 1 was set to 0.2. A trade off for γ exists such that a higher value will result in free space being correctly classified, while certain regions on static structures like buildings incorrectly classified as free space. While a lower value will result in static structures likely to be correctly inferred to be on occupied space, but certain pixels on roads incorrectly classified as on occupied space. We set the threshold small as in our experiments, we found that preserving the edges of buildings is particularly important in ensuring the pseudo point-cloud captures geometric cues also observed in lidar scans.

Finally, the design parameters in our networks, such as the descriptor dimension, network width and depth, and the number of heads for multi-head attention in the Transformer module were chosen by trail and error to achieve the best localisation accuracy.

4.6.3 Training Details

The training takes three stages. We first pretrain f_o for 100 epochs using the loss in Equation (5), 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 (7). Finally, we fix the network parameters of f_o and DCP and optimise a PointNetVLAD layer for place recognition using the bidirectional triplet loss in Equation (14). While in theory it is possible to optimise DCP and PointNetVLAD for registration and place recognition jointly, empirically this resulted in larger metric localisation errors. As such, we resorted to training the three stages sequentially. We use a fixed learning rate of 2×10^{-4} in stages 1 and 3, and 1×10^{-4} in stage 2. We use Adam as the optimiser for all experiments. We train stage 2 for 160 epochs and stage 3 for 120 epochs, and choose the check-point with the best validation set performance.

5 Experimental Results

We validate our method using the Oxford Radar RobotCar Dataset [59] and the KITTI Raw Dataset [60]. The lidar data are collected using Velodyne HDL-32E for the first and Velodyne HDL-64E for the latter. Both datasets have longitude/latitude data for each time-stamp to query 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. The parts of KITTI data we use were recorded in 2011, while the Oxford Radar RobotCar Dataset was collected in 2019. The satellite data for both datasets were queried in 2021, years after the collection of the corresponding lidar data. Our method achieves decent localisation accuracy despite potential geometry changes in the scene during this time span.

Our method is the first to learn the place recognition of a lidar using only overhead imagery. Thus, we benchmark against a baseline method that trains a VGG-16 network [61] with BatchNorm followed by a NetVLAD [44] layer directly on the lidar and overhead images, using triplet loss. We compare our pose estimation approach against recent work [13] on learning the metric localisation between range sensors and satellite imagery. Finally, we compare against our prior work in [16]. To the best of our knowledge, the work in [13–16] are the only methods that solve single frame lidar-only localisation against overhead imagery, and evaluated on the datasets we use.

For a fair comparison, our choice of training, validation, and test trajectories in each dataset is made as close to the lidar experiments in [14] as possible. Specifically, 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.

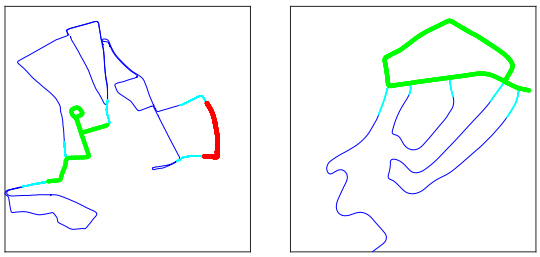


Fig. 8: Trajectories are split into training (blue), validation (red), and test (green) for all sequences in RobotCar (left) and `20111003_drive0034` in KITTI (right). We discard regions near the intersections between splits (cyan) to avoid any overlap between training, validation, and test trajectories.

5.1 Metric Localisation

To evaluate metric localisation, we assume place recognition is solved such that we have spatially

Initial offset for x, y, and θ (pixel, pixel, °)	±25 x	±25 y	±180 θ	±10 x	±10 y	±180 θ	±25 x	±25 y	±90 θ
RSL-Net [13]	**7.45**	**9.51**	68.53	6.08	8.05	58.69	12.45	13.03	34.79
Tang et al. [16]	9.77	10.28	49.18	4.84	4.77	25.36	7.43	10.18	11.41
Ours	7.52	9.94	**28.89**	**4.35**	**4.31**	**18.99**	**7.19**	**8.81**	**10.43**
Initial offset for x, y, and θ (pixel, pixel, °)	±10 x	±10 y	±90 θ	±25 x	±25 y	±45 θ	±25 x	±25 y	±22.5 θ
RSL-Net [13]	5.90	6.81	10.75	5.85	**7.04**	**2.36**	**5.33**	**5.89**	**2.08**
Tang et al. [16]	4.39	4.13	7.16	6.34	7.85	5.53	5.89	7.54	3.50
Ours	**3.93**	**4.00**	**5.70**	**5.84**	7.43	5.14	5.49	6.72	4.05

Table 2: 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, °)	±25 x	±25 y	±180 θ	±10 x	±10 y	±180 θ	±25 x	±25 y	±90 θ
RSL-Net [13]	7.29	11.93	70.93	5.33	7.45	67.28	**5.62**	9.49	11.19
Tang et al. [16]	**6.91**	11.22	62.79	**3.71**	4.50	34.51	5.89	8.89	**9.10**
Ours	7.04	**10.65**	**58.92**	3.87	**4.35**	**26.50**	5.79	**8.48**	9.82
Initial offset for x, y, and θ (pixel, pixel, °)	±10 x	±10 y	±90 θ	±25 x	±25 y	±45 θ	±25 x	±25 y	±22.5 θ
RSL-Net [13]	4.56	7.61	8.36	5.54	9.93	**2.56**	5.34	**6.08**	**1.59**
Tang et al. [16]	**3.44**	4.41	6.73	5.14	6.97	4.52	5.08	7.15	3.70
Ours	3.49	**3.67**	**3.82**	**4.74**	**6.93**	4.23	**4.87**	6.60	3.36

Table 3: 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, °)	±25 x	±25 y	±180 θ	±10 x	±10 y	±180 θ	±25 x	±25 y	±90 θ
RSL-Net [13]	**7.11**	**8.49**	62.78	**3.27**	4.69	59.04	7.86	**8.27**	**8.51**
Tang et al. [16]	8.47	11.38	49.62	4.86	5.38	35.90	7.41	9.55	17.02
Ours	7.81	11.37	**43.29**	4.43	**4.65**	**25.64**	**7.24**	10.26	12.83
Initial offset for x, y, and θ (pixel, pixel, °)	±10 x	±10 y	±90 θ	±25 x	±25 y	±45 θ	±25 x	±25 y	±22.5 θ
RSL-Net [13]	**3.43**	4.30	13.73	7.17	8.48	**3.80**	6.47	**7.45**	**2.90**
Tang et al. [16]	4.10	4.64	13.00	6.25	**8.07**	7.44	6.55	7.78	5.59
Ours	3.78	**4.22**	**5.28**	**6.04**	8.63	6.62	**6.03**	7.89	5.31

Table 4: 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.

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.

DCP [38] includes a Transformer [56] module for learning local descriptors, which uses multi-headed attention. In our prior work [16], the descriptor length d was set to 512, while the number of heads was 4. Here we use an improved architecture where we increase d to 1024 and the number of heads to 16. We also introduce additional random rotations to the input point-clouds for data augmentation when training the DCP network. We show that, using an improved architecture and rotation augmentation, the metric localisation results consistently outperform our prior work.

We compare against RSL-Net [13] and our prior work [16] where the initial pose offset is a

uniform distribution in each of x, y, and θ within a specific 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 2 and 3.

In the original work [13, 14], RSL-Net was originally evaluated on experiments with small rotation offsets 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 our prior work [16], 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. Using the improved architecture with larger feature dimension and number of heads in the Transformer module, the forward pass now takes roughly 0.1 s, having approximately the same run-time performance as RSL-Net.

5.1.1 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 actual offset as such a solution is equally "correct" for symmetrical point-clouds. The point-cloud registration problem is ill-posed if symmetry exists, in particular if the initial rotation offset can be anywhere from $-\pi$ to π.

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

$$\mathrm{d}_{\mathrm{chamfer}} = \sum_i \mathrm{d}_{\mathrm{min}}(\bar{\mathbf{p}}_i^S, \mathbf{R}_\pi \bar{\mathbf{P}}^S), \qquad (15)$$

where $\bar{\mathbf{P}}^S$ is $\mathbf{P}^S$ centred about its centroid, and each $\bar{\mathbf{p}}_i^S \in \mathbb{R}^2$ is a point in $\bar{\mathbf{P}}^S$, $\mathrm{d}_{\mathrm{min}}(\mathbf{p}, \mathbf{P})$ 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 (15) 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}.



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

5.2 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 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 $[-5\ \mathrm{m}, 5\ \mathrm{m}]$ to each of the x and y directions 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

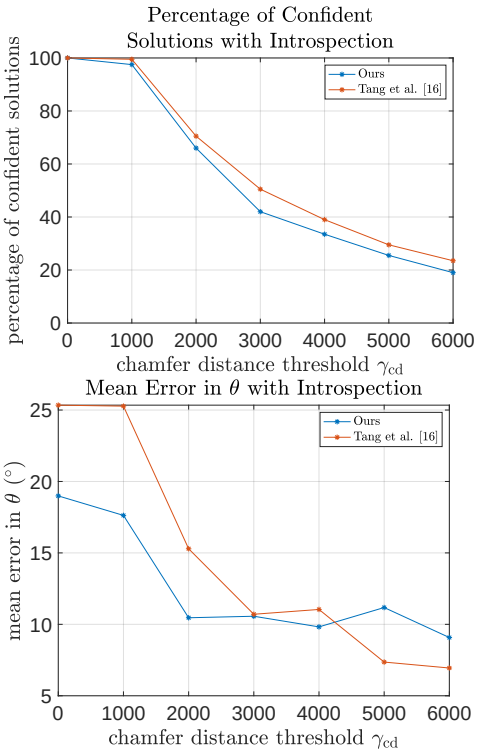


Fig. 10: Percentage of solutions considered "confident" and the mean θ error of confident solutions, for various values of γ_{cd}, evaluated on RobotCar.

the vehicle will likely face the same (or 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. In fact, without another sensor (e.g., a magnetometer) to measure the global heading, spatially proximal lidar and overhead image pairs can 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 $\mathbf{P}^L$ and $\mathbf{P}^{L+}$ and $\mathbf{P}^S$ when training the PointNetVLAD layer.

5.2.1 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

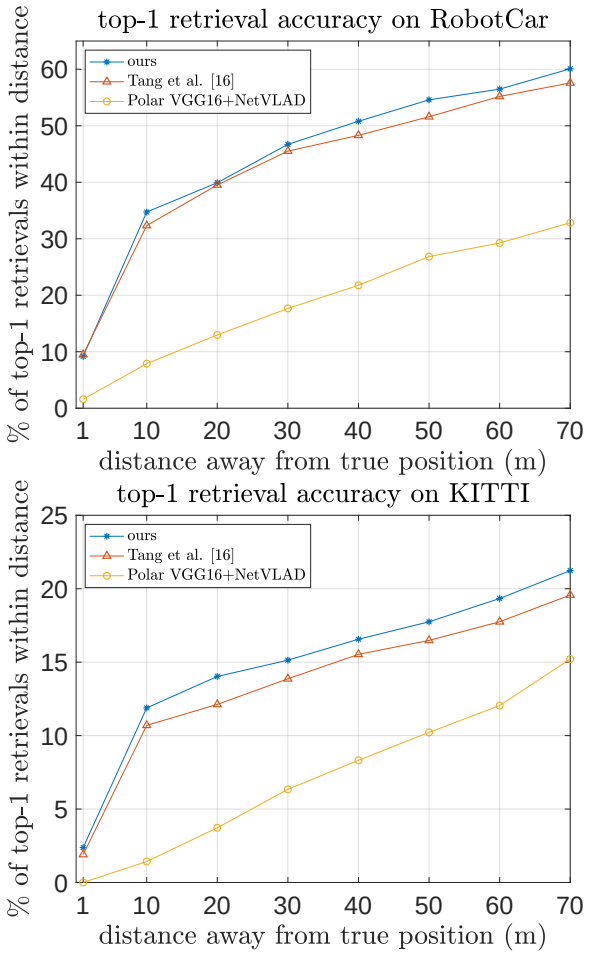


Fig. 11: Percentage of top-1 retrievals falling within a certain distance away from the true position, evaluated on RobotCar (top) and KITTI (bottom).

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 matches within certain distances away from the true position. We show that using the new bidirectional triplet loss in Equation (14), our results are superior to our prior work [16], which uses the standard triplet loss in Equation (13). We also 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 inputs as polar images before passing them 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 40 m of the true position. The test set for KITTI, featuring a residential area, is much

more challenging as residential areas are more structurally repetitive and have fewer geometrically distinctive places. Regardless, our method consistently outperforms our prior work and the baseline method.

5.2.2 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 12.

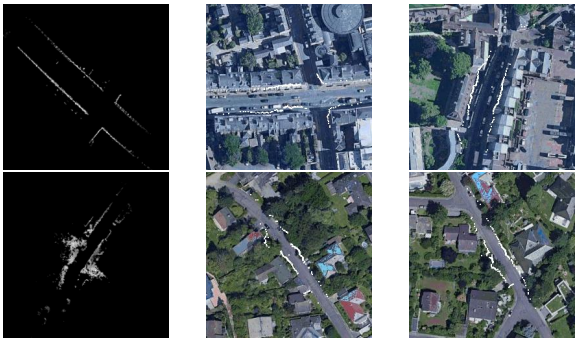


Fig. 12: Failure cases: a lidar image I^L_j along the test route (left), the satellite image and the associated pseudo point-cloud corresponding to the true position (centre), and the falsely retrieved satellite image and its associated pseudo point-cloud (right), evaluated on RobotCar (top) and KITTI (bottom).

5.2.3 Precision and Recall

To evaluate precision and recall, we sample a threshold γ_{desc} uniformly from the minimum Euclidean distance between any $\mathbf{d}^S_i$ to any $\mathbf{d}^L_j$, to the maximum. A pair $\mathbf{d}^S_i$ and $\mathbf{d}^L_j$ 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 our prior work [16] and the baseline as shown in Figure 13.

5.2.4 Descriptor Smoothing Using a Sequence of Data

The results presented in Section 5.2.1 use a single frame of lidar data for retrieval. However, the accuracy is corrupted by false retrievals caused by structurally similar regions as shown

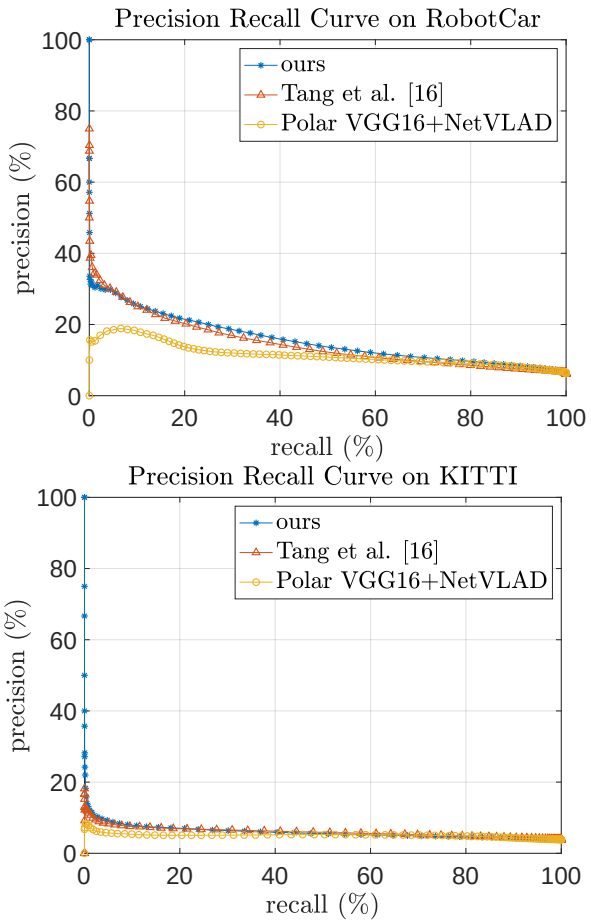


Fig. 13: Precision and recall curve for various descriptor distance thresholds.

in Section 5.2.2. A simple yet surprisingly effective amendment is to use a sequence of lidar data, rather than a single frame, at inference time. Specifically, given the ith lidar image at inference time, I^L_i, we can form a set of lidar images by finding its K nearest neighbours (time-wise) in the lidar data stream, namely

$$\Omega^L_i = \{\mathrm{I}^L_{i-\frac{K}{2}}, \mathrm{I}^L_{i-\frac{K}{2}+1}, \ldots, \mathrm{I}^L_i, \ldots, \mathrm{I}^L_{i+\frac{K}{2}}\}, \quad (16)$$

and their associated global descriptors

$$\Phi^L_i = \{\mathbf{d}^L_{i-\frac{K}{2}}, \mathbf{d}^L_{i-\frac{K}{2}+1}, \ldots, \mathbf{d}^L_i, \ldots, \mathbf{d}^L_{i+\frac{K}{2}}\}. \quad (17)$$

Since satellite and roadmap images are also queried along a known route, given the jth pair of images I^S_j and I^R_j, we can also find a sequence of global descriptors, forming the set

$$\Phi^S_i = \{\mathbf{d}^S_{i-\frac{K}{2}}, \mathbf{d}^S_{i-\frac{K}{2}+1}, \ldots, \mathbf{d}^S_i, \ldots, \mathbf{d}^S_{i+\frac{K}{2}}\}. \quad (18)$$

We can then take the median over all $K+1$ descriptors from the set:

$$\tilde{\mathbf{d}}_i^L = \text{median}(\Phi_i^L), \qquad \tilde{\mathbf{d}}_j^S = \text{median}(\Phi_j^S), \quad (19)$$

and use $\tilde{\mathbf{d}}_i^L$ and $\tilde{\mathbf{d}}_j^S$ for retrieval, effectively *smoothing* the descriptors with a stream of data. The resulting retrieval accuracies for different values of K are shown in Figure 14. We can achieve significantly higher accuracy by using a stream of data. For $K = 40$, on the RobotCar dataset, almost 75% of all top-1 retrievals fall within 70 m of the true position.

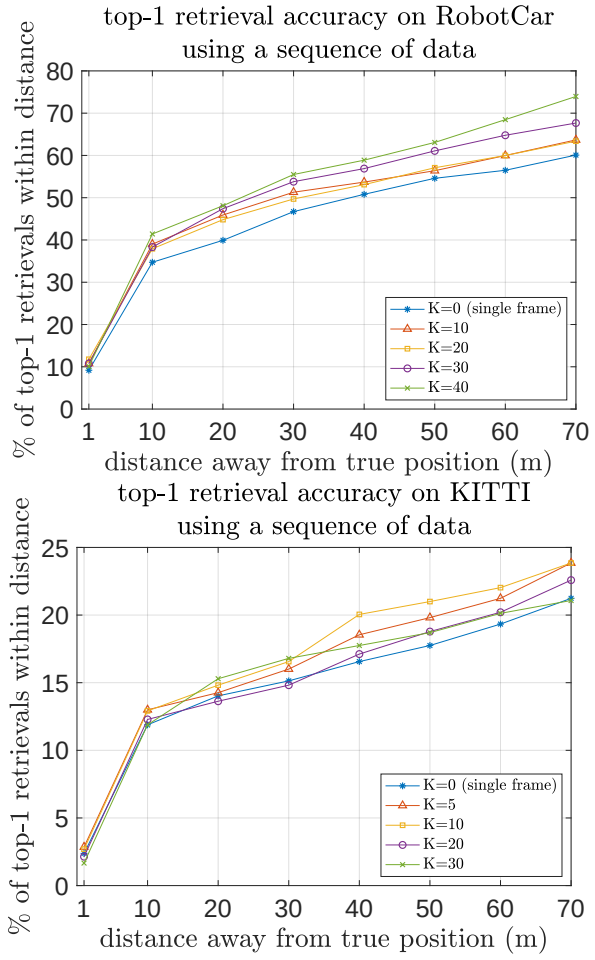


Fig. 14: Percentage of top-1 retrievals falling within a certain distance away from the true position, evaluated on RobotCar (top) and KITTI (bottom), by applying descriptor smoothing using a sequence of data.

Given a specific value of K, all $\tilde{\mathbf{d}}_j^S$ can be computed and stored in an offline process. However, for a live lidar data stream, a minimum of $\frac{K}{2}$ frames must be delayed before $\tilde{\mathbf{d}}_i^L$ can be computed for the ith frame, making such approach impracticable when delays are not acceptable. Nevertheless, the top-1 retrieval accuracies are consistently better than using single frames of data only. We have also experimented with taking the mean for smoothing rather than the median. However, this resulted in slightly lower accuracies than the median. Vitally, no additional training or retraining is needed to apply descriptor smoothing at inference time.

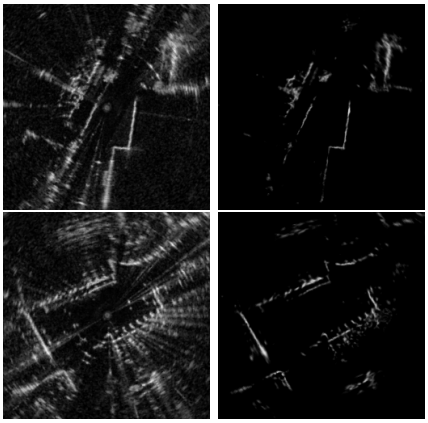


Fig. 15: Radar images (top left and bottom left) transformed into the appearance of lidar images (top right and bottom right) using the CycleGAN implementation in [62].

5.3 Extending to Radar Localisation using Overhead Imagery

Our method relies on the premise that pixels from the centroid to the first return along each azimuth are considered free-space in a lidar image. This assumption is not valid for scanning radar images that are prone to speckle interference. However, we can reliably transform radar images to the appearance of lidar images using unpaired image-to-image transfer [51], specifically the methodology in [62].

Figure 15 shows some synthetic lidar images created from radar images. Once radar images are transformed into the appearance of lidar images, they are used in this form as input in our method for training and inference. We evaluate on the RobotCar Dataset, which also has data from a Navtech radar, and use the same sequences and

Initial offset for x, y, and θ (pixel, pixel, °)	±25 x	±25 y	±180 θ	±10 x	±10 y	±180 θ	±25 x	±25 y	±90 θ
sequence no.2	7.52	9.94	**28.89**	4.35	4.31	18.99	7.19	8.81	10.43
sequence no.7	**6.75**	**9.29**	33.10	4.41	**3.72**	16.89	7.84	**6.20**	**9.04**
sequence no.18	9.11	10.17	33.68	**4.16**	4.29	**14.47**	**6.90**	8.25	10.39
Initial offset for x, y, and θ (pixel, pixel, °)	±10 x	±10 y	±90 θ	±25 x	±25 y	±45 θ	±25 x	±25 y	±22.5 θ
sequence no.2	3.93	**4.00**	5.70	5.84	7.43	5.14	5.49	6.72	**4.05**
sequence no.7	**3.72**	4.03	**5.02**	**5.33**	6.76	**4.73**	**5.23**	**5.95**	4.31
sequence no.18	3.81	4.54	7.37	6.13	**6.55**	5.58	5.89	6.07	4.22

Table 5: Lidar metric localisation errors for various ranges of initial offset, evaluated on RobotCar for various test set sequences. The units are pixels for x and y and degrees for θ.

Dataset	RobotCar			KITTI			RobotCar (radar)		
	x	y	θ	x	y	θ	x	y	θ
RSL-Net [13]	6.08	8.05	58.69	5.33	7.45	67.28	**3.27**	4.69	59.04
Tang et al. [16]	4.84	4.77	25.36	**3.71**	4.50	34.51	4.86	5.38	35.90
ours (every 10th)	**4.14**	4.32	22.45	3.80	4.59	36.29	4.58	4.90	32.73
ours (first 2000)	4.21	5.44	32.36	3.89	4.67	49.28	4.86	5.00	36.12
ours (full)	4.35	**4.31**	**18.99**	3.87	**4.35**	**26.50**	4.43	**4.65**	**25.64**

Table 6: Metric localisation errors for an initial offset of ±10 pixels, ±10 pixels, ±180°, evaluated on various datasets for ablation studies. The units are pixels for x and y and degrees for θ.

splits as in the lidar experiments. The metric localisation and place recognition results are shown in Table 4 and Figure 16.

5.4 Testing on Different Sequences

So far, the experiments presented are evaluated on test set trajectories collected in places unseen from the training set, as illustrated in Figure 8. However, the training and test data are still from the same day. The RobotCar Dataset features repeated traversals of the same route over different days. The recorded on-board lidar data can have minor variations over the same route across different days, for example due to different non-static objects like cars or pedestrians, or different driving patterns depending on the traffic condition.

Here we demonstrate that our method can not only generalise to unseen places, but also to data collected across different times.

Our training data come from sequences no.2, no.5, and no.6. For additional validation we arbitrarily selected the test set trajectories from sequences no.7 and no.18, which are collected 1 day and 5 days after the training data, respectively. The results for metric localisation are shown in Table 5 and compared to the original results presented in Table 2, which uses the test set trajectory in sequence no.2. The mean errors for metric localisation do not change significantly across different sequences.

Figure 17 shows place recognition results evaluated on sequences no.7 and no.18. The top-1 retrieval accuracy drops slightly when evaluated on different sequences, but does not degrade too significantly.

5.5 Ablation Studies

The training data contain approximately 14 thousand images for each dataset, where the data are prepared as detailed in Section 5 and Figure 8. Here we perform ablation studies by only using a fraction of training data. The first reduced training set is formed by uniformly sampling 10% of the training data by selecting every 10th frame. The second reduced training set is formed by selecting the first 2000 images in the original training set. Models are trained for each reduced training set, where the number of training epochs is kept the same as using the full dataset.

The results for metric localisation under an initial offset of ±10 pixels, ±10 pixels, ±180° for x, y, and θ are shown in Table 6. The results on top-1

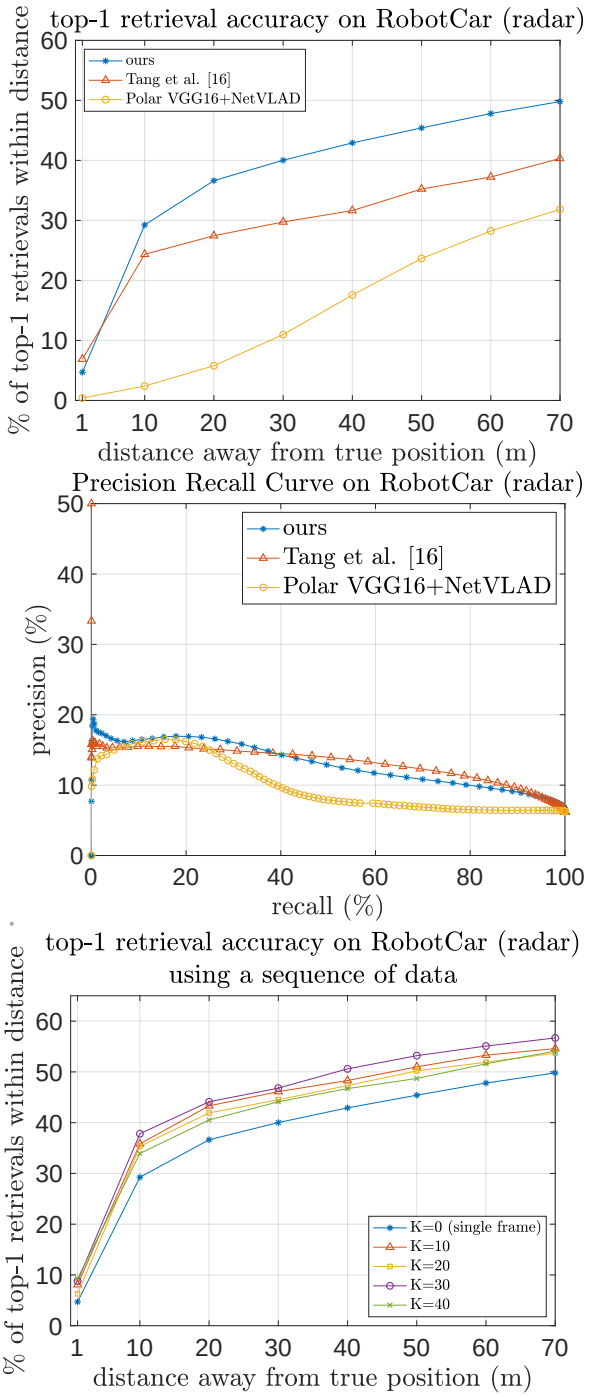


Fig. 16: Place recognition results using radar, evaluated on the RobotCar Dataset. Top: top-1 retrieval accuracy. Centre: precision and recall curve. Bottom: top-1 retrieval accuracy with descriptor smoothing

retrieval accuracy for place recognition are shown in Figure 18.

Shown in Table 6, even with reduced training data, our method resulted in smaller metric localisation errors than RSL-Net [13]. Place recognition results are also greatly superior than the

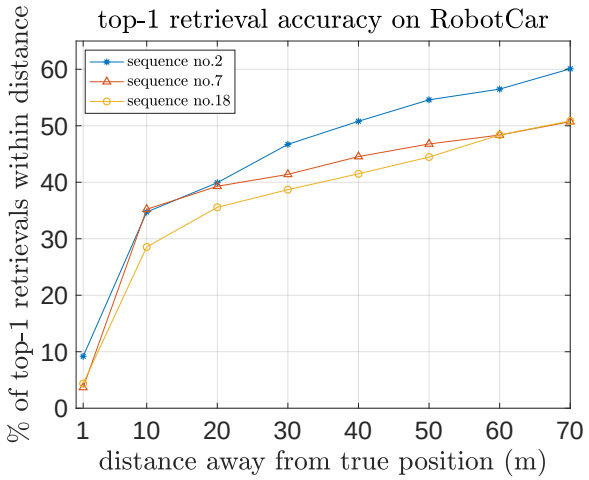


Fig. 17: Place recognition retrieval accuracies on the RobotCar dataset for lidar localisation, evaluated for different sequences.

VGG16+NetVLAD baseline method even when our method is trained with reduced training data. Furthermore, though the number of data samples is larger when we select the first 2000 frames than every 10th frame, uniformly sampling every 10th frame forms a training set with greater variation, thus resulting in better performance on test data.

5.6 Additional Qualitative Results

Additional qualitative results are shown in Figure 19.

6 Conclusion and Future Work

This article presents a method that solves both place recognition and metric localisation of lidar using only publicly available overhead imagery. Specifically, by inferring occupancy information from overhead imagery as an intermediate representation, the modality difference is handled naturally where both data sources are represented as point sets. While our method is the first to learn the place recognition of a ground lidar using overhead imagery, it also outperforms the prior image-based methods on the metric localisation between lidar and overhead imagery when the initial pose offset is large. While the method as detailed in this paper is standalone and requires only overhead imagery, future work would lead to the integration within a large-scale lidar localisation system with prior lidar maps, where solutions using overhead imagery are used as auxiliary signals to improve

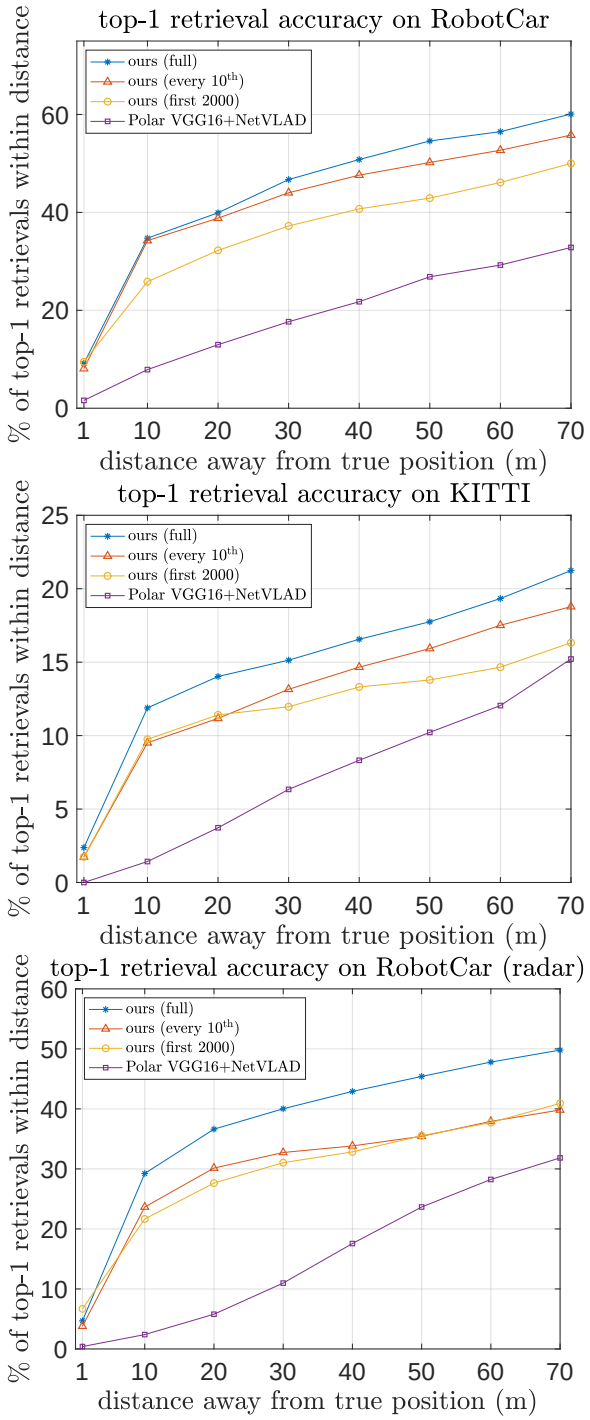
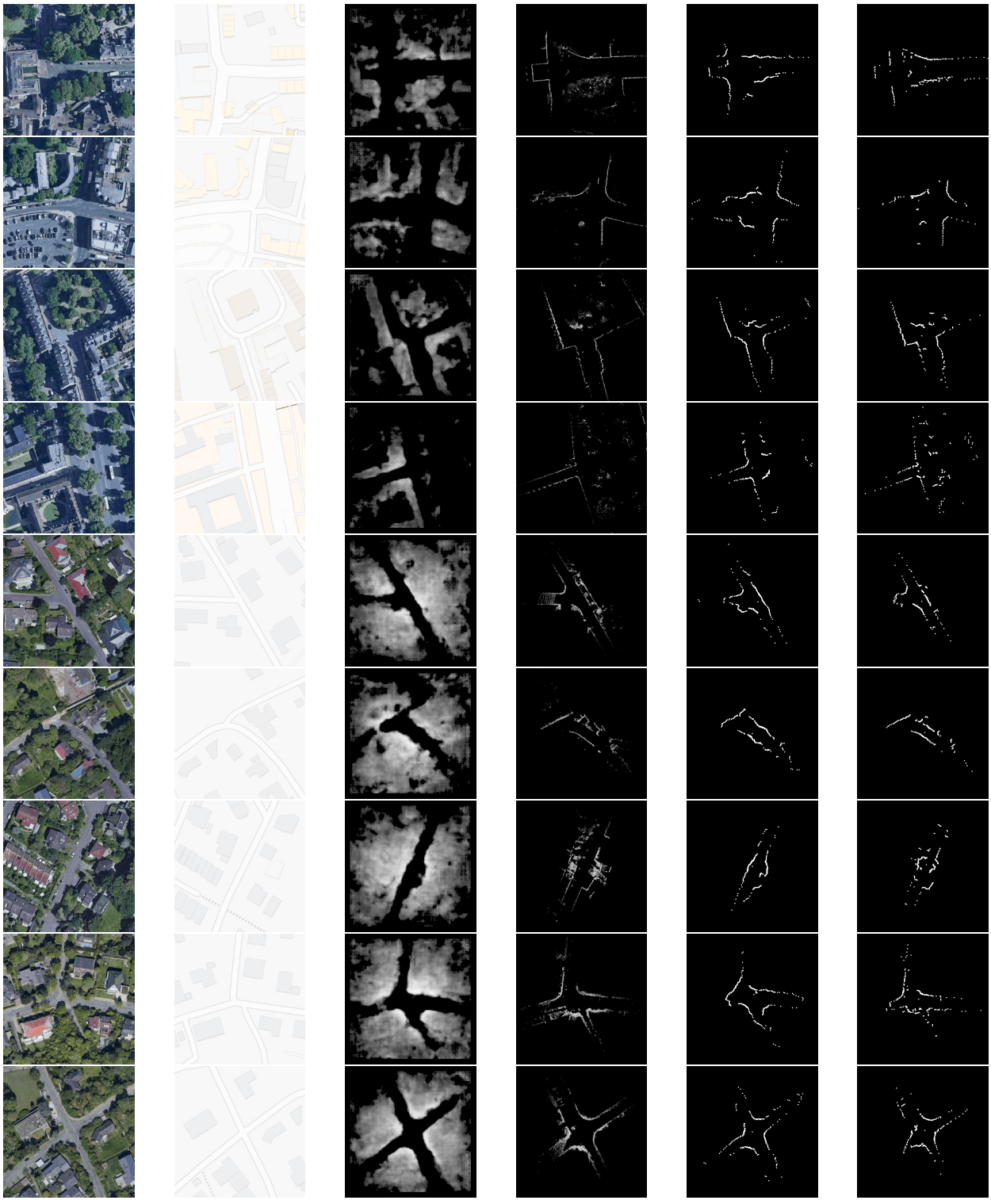


Fig. 18: Percentage of top-1 retrievals within a certain distance away from the true position, evaluated on RobotCar (top), KITTI (centre), and RobotCar with radar localisation (bottom), for ablation studies.

accuracy further. An interesting study would be to take the DCP network previously trained on one dataset, and train PointNetVLAD for place recognition on another dataset, so that place recognition can be learned for the new dataset without needing to re-learn metric localisation.

Another interesting research direction would be to learn the place recognition between a ground range sensor and overhead imagery in an unsupervised fashion, without paired ground truth data.



(a) (b) (c) (d) (e) (f)

Fig. 19: Qualitative results from the test set of RobotCar (rows 1 to 4) and KITTI (rows 5 to 9). 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.

References

[1] Bosse, M., Zlot, R.: Place Recognition Using Keypoint Voting in Large 3D Lidar Datasets. In: 2013 IEEE International Conference on Robotics and Automation, pp. 2677–2684

(2013). IEEE

[2] Dubé, R., Dugas, D., Stumm, E., Nieto, J., Siegwart, R., Cadena, C.: Segmatch: Segment Based Place Recognition in 3d Point Clouds. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 5266–5272 (2017). IEEE

[3] Kim, G., Kim, A.: 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), pp. 4802–4809 (2018). IEEE

[4] Guo, J., Borges, P.V., Park, C., Gawel, A.: Local Descriptor for Robust Place Recognition Using Lidar Intensity. IEEE Robotics and Automation Letters **4**(2), 1470–1477 (2019)

[5] Wolcott, R.W., Eustice, R.M.: Fast LIDAR Localization Using Multiresolution Gaussian Mixture Maps. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 2814–2821 (2015). IEEE

[6] Barsan, I.A., Wang, S., Pokrovsky, A., Urtasun, R.: Learning to Localize Using a LiDAR Intensity Map. In: CoRL, pp. 605–616 (2018)

[7] Baldwin, I., Newman, P.: Road Vehicle Localization with 2D Push-broom LIDAR and 3D Priors. In: 2012 IEEE International Conference on Robotics and Automation, pp. 2611–2617 (2012). IEEE

[8] Yew, Z.J., Lee, G.H.: 3DFeat-Net: Weakly Supervised Local 3D Features for Point Cloud Registration. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 607–623 (2018)

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

[10] Cop, K.P., Borges, P.V., Dubé, R.: Delight: An Efficient Descriptor for Global Localisation Using lidar Intensities. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 3653–3660 (2018). IEEE

[11] Yin, H., Wang, Y., Ding, X., Tang, L., Huang, S., Xiong, R.: 3D Lidar-Based Global Localization Using Siamese Neural Network. IEEE Transactions on Intelligent Transportation Systems **21**(4), 1380–1392 (2019)

[12] Sun, L., Adolfsson, D., Magnusson, M., Andreasson, H., Posner, I., Duckett, T.: Localising Faster: Efficient and Precise Lidar-Based Robot Localisation in Large-Scale Environments. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 4386–4392 (2020). IEEE

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

[14] Tang, T., Martini, D.D., Wu, S., Newman, P.: Self-Supervised Localisation between Range Sensors and Overhead Imagery. In: Proceedings of Robotics: Science and Systems, Corvalis, Oregon, USA (2020). https://doi.org/10.15607/RSS.2020.XVI.057

[15] Tang, T.Y., De Martini, D., Wu, S., Newman, P.: Self-supervised learning for using overhead imagery as maps in outdoor range sensor localization. The International Journal of Robotics Research **40**(12-14), 1488–1509 (2021)

[16] Tang, T.Y., De Martini, D., Newman, P.: Get to the Point: Learning Lidar Place Recognition and Metric Localisation Using Overhead Imagery. In: Proceedings of Robotics: Science and Systems, Virtual (2021). https://doi.org/10.15607/RSS.2021.XVII.003

[17] Leung, K.Y.K., Clark, C.M., Huissoon, J.P.: Localization in Urban Environments by Matching Ground Level Video Images with an Aerial Image. In: 2008 IEEE International

Conference on Robotics and Automation, pp. 551–556 (2008). IEEE

[18] Noda, M., Takahashi, T., Deguchi, D., Ide, I., Murase, H., Kojima, Y., Naito, T.: Vehicle Ego-Localization by Matching in-Vehicle Camera Images to an Aerial Image. In: Asian Conference on Computer Vision, pp. 163–173 (2010). Springer

[19] Senlet, T., Elgammal, A.: 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), pp. 2034–2041 (2011). IEEE

[20] Pink, O.: Visual Map Matching and Localization Using a Global Feature Map. In: 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 1–7 (2008). IEEE

[21] Kim, D.-K., Walter, M.R.: Satellite Image-Based Localization via Learned Embeddings. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 2073–2080 (2017). IEEE

[22] Chu, H., Mei, H., Bansal, M., Walter, M.R.: Accurate Vision-Based Vehicle Localization Using Satellite Imagery. arXiv preprint arXiv:1510.09171 (2015)

[23] de Paula Veronese, L., de Aguiar, E., Nascimento, R.C., Guivant, J., Cheein, F.A.A., De Souza, A.F., Oliveira-Santos, T.: 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), pp. 4285–4290 (2015). IEEE

[24] Kümmerle, R., Steder, B., Dornhege, C., Kleiner, A., Grisetti, G., Burgard, W.: Large Scale Graph-based SLAM using Aerial Images as Prior Information. Autonomous Robots **30**(1), 25–39 (2011)

[25] Dogruer, C.U., Koku, A.B., Dolen, M.: Outdoor Mapping and Localization Using Satellite Images. Robotica **28**(7), 1001–1012 (2010)

[26] Hussein, M., Renner, M., Watanabe, M., Iagnemma, K.: 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, pp. 1432–1437 (2013). IEEE

[27] Carle, P.J., Barfoot, T.D.: Global Rover Localization by Matching Lidar and Orbital 3D Maps. In: 2010 IEEE International Conference on Robotics and Automation, pp. 881–886 (2010). IEEE

[28] Zhu, M., Yang, Y., Song, W., Wang, M., Fu, M.: AGCV-LOAM: air-ground cross-view based LiDAR odometry and mapping. In: 2020 Chinese Control And Decision Conference (CCDC), pp. 5261–5266 (2020). IEEE

[29] Fervers, F., Bullinger, S., Bodensteiner, C., Arens, M., Stiefelhagen, R.: Continuous Self-Localization on Aerial Images Using Visual and Lidar Sensors. arXiv preprint arXiv:2203.03334 (2022)

[30] Brubaker, M.A., Geiger, A., Urtasun, R.: Lost! Leveraging the Crowd for Probabilistic Visual Self-localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3057–3064 (2013)

[31] Floros, G., Van Der Zander, B., Leibe, B.: OpenStreetSLAM: Global Vehicle Localization Using OpenStreetMaps. In: 2013 IEEE International Conference on Robotics and Automation, pp. 1054–1059 (2013). IEEE

[32] Panphattarasap, P., Calway, A.: 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), pp. 6341–6348 (2018). IEEE

[33] Yan, F., Vysotska, O., Stachniss, C.: Global Localization on OpenStreetMap Using 4-bit Semantic Descriptors. In: 2019 European Conference on Mobile Robots (ECMR), pp.

1–7 (2019). IEEE

[34] Besl, P.J., McKay, N.D.: Method for Registration of 3-D Shapes. In: Sensor Fusion IV: Control Paradigms and Data Structures, vol. 1611, pp. 586–606 (1992). International Society for Optics and Photonics

[35] Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T.: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1802–1811 (2017)

[36] Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. arXiv preprint arXiv:1706.02413 (2017)

[37] Choy, C., Dong, W., Koltun, V.: Deep Global Registration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2514–2523 (2020)

[38] Wang, Y., Solomon, J.M.: Deep Closest Point: Learning Representations for Point Cloud Registration. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3523–3532 (2019)

[39] Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic Graph CNN for Learning on Point Clouds. Acm Transactions On Graphics (tog) **38**(5), 1–12 (2019)

[40] Wang, Y., Solomon, J.M.: PRNet: Self-Supervised Learning for Partial-to-Partial Registration. arXiv preprint arXiv:1910.12240 (2019)

[41] Yuan, W., Eckart, B., Kim, K., Jampani, V., Fox, D., Kautz, J.: DeepGMR: Learning Latent Gaussian Mixture Models for Registration. In: European Conference on Computer Vision, pp. 733–750 (2020). Springer

[42] Uy, M.A., Lee, G.H.: PointNetVLAD: Deep Point Cloud Based Retrieval for Large-Scale Place Recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4470–4479 (2018)

[43] Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 652–660 (2017)

[44] Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5297–5307 (2016)

[45] Liu, Z., Zhou, S., Suo, C., Yin, P., Chen, W., Wang, H., Li, H., Liu, Y.-H.: 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, pp. 2831–2840 (2019)

[46] Zhang, W., Xiao, C.: 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, pp. 12436–12445 (2019)

[47] Du, J., Wang, R., Cremers, D.: DH3D: Deep Hierarchical 3D Descriptors for Robust Large-Scale 6DoF Relocalization. In: European Conference on Computer Vision, pp. 744–762 (2020). Springer

[48] Barnes, D., Posner, I.: 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), pp. 9484–9490 (2020). IEEE

[49] Chen, X., Läbe, T., Milioto, A., Röhling, T., Vysotska, O., Haag, A., Behley, J., Stachniss, C.: OverlapNet: Loop Closing for LiDAR-based SLAM. In: Proceedings of Robotics: Science and Systems, Corvalis, Oregon, USA (2020). https://doi.org/10.15607/RSS.2020.XVI.009

[50] Isola, P., Zhu, J.-Y., Zhou, T., Efros, A.A.: Image-to-Image Translation with Conditional Adversarial Networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1125–1134 (2017)

[51] Zhu, J.-Y., Park, T., Isola, P., Efros, A.A.: Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2223–2232 (2017)

[52] Lenc, K., Vedaldi, A.: Understanding Image Representations by Measuring Their Equivariance and Equivalence. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 991–999 (2015)

[53] Weston, R., Cen, S., Newman, P., Posner, I.: Probably Unknown: Deep Inverse Sensor Modelling Radar. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 5446–5452 (2019). IEEE

[54] Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. In: International Conference on Medical Image Computing and Computer-assisted Intervention, pp. 234–241 (2015). Springer

[55] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

[56] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all You Need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 6000–6010 (2017)

[57] Sorkine, O.: Least-squares rigid motion using svd. Technical notes **120**(3), 52 (2009)

[58] Chen, W., Chen, X., Zhang, J., Huang, K.: Beyond triplet loss: a deep quadruplet network for person re-identification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 403–412 (2017)

[59] Barnes, D., Gadd, M., Murcutt, P., Newman, P., Posner, I.: The Oxford Radar Robotcar dataset: A Radar Extension to the Oxford Robotcar Dataset. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 6433–6438 (2020). IEEE

[60] Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision Meets Robotics: The KITTI Dataset. International Journal of Robotics Research (IJRR) (2013)

[61] Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition (2015)

[62] Weston, R., Jones, O.P., Posner, I.: There and Back Again: Learning to Simulate Radar Data for Real-World Applications. arXiv preprint arXiv:2011.14389 (2020)

6

Self-Supervised Lidar Place Recognition in Overhead Imagery Using Unpaired Data

This is an ongoing work and draft article in preparation to be submitted as a conference proceeding to Robotics: Science and System XIV for peer review in February 2023.

Background and Motivation

All methods presented in Chapters 3 to 5 require *paired* range sensor data and overhead imagery for training. This in turn requires an on-board GPS for global position information when collecting training data – though the position information needs not to have sub-metre level accuracy in the case of our work in Chapter 4. In real life applications, the cost and labour of training data creation can be greatly reduced if the need for on-board, time-synchronised global position information can be alleviated.

Motivated by this, we propose a novel method to learn the cross-modal retrieval (and thus place recognition) between range sensor data and overhead imagery in a self-supervised fashion, entirely from *unpaired* data.

Self-Supervised Lidar Place Recognition in Overhead Imagery Using Unpaired Data

Author Names Omitted for Anonymous Review. Paper-ID [add your ID here]

***Abstract*—As much as place recognition is crucial for navigation, mapping and collecting training ground truth, namely sensor data pairs across different locations, are costly and time-consuming. This paper tackles both issues by learning lidar place recognition on publicly available overhead imagery and in a self-supervised fashion, with no need for paired lidar and overhead imagery data. We learn the cross-modal data comparison between lidar and overhead imagery with a multi-step framework. First, images are transformed into synthetic lidar data and a latent projection is learned. Next, we discover pseudo pairs of lidar and satellite data from unpaired and unsynchronised sequences. We then refine such projection by using the discovered pairs to learn a cross-modality place recognition framework. We train and test our approach on real data from different environments and show performances approaching prior fully-supervised methods.**

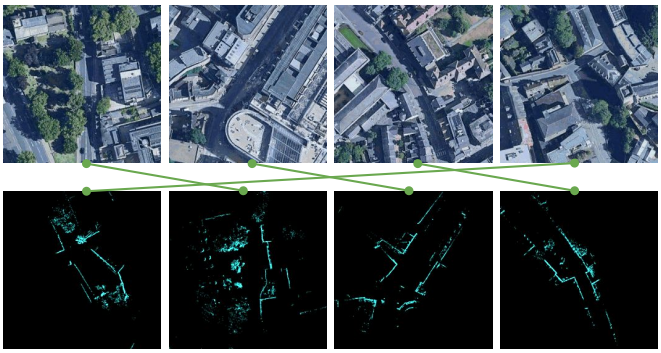


Fig. 1: With paired data, we can sample positive pairs of satellite and lidar images, connected by green lines. This information is unavailable in unpaired data. All images are rotation-aligned for better visualisation.

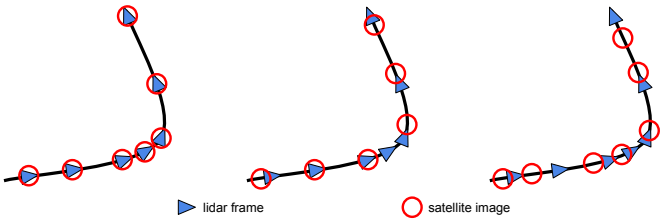


Fig. 2: Lidar data are collected at a certain frequency as a vehicle travels along a route. **Left:** in the supervised case, a satellite image is queried at the centre position of each lidar scan, forming one-to-one corresponding pairs. **Middle:** satellite images are sampled uniformly along the route, resulting in unpaired data. **Right:** satellite images are sampled based on unsynchronised position measurements from a different traversal, resulting in unpaired data.

I. Introduction

Lidar is widely considered an ideal sensor for outdoor operation, as it provides a long sensing range, 360° field-of-view (FOV), invariance to lighting, and robustness against weather conditions. For this reason, place recognition, also known as topological localisation, has been extensively studied for lidar [5, 15, 28, 20, 8, 41, 62]. Existing methods require lidar data to have been previously recorded in the places of operation, either as an aggregated point-cloud map or individual scans. In scenarios where prior sensor data become unreliable or are unavailable, off-the-shelf overhead imagery such as Google satellite images can be used as an alternative data source for place recognition. Even under normal operating conditions, overhead imagery can serve as an additional information source for redundancy.

When projected to the x-y plane and expressed as a 2D scan image, lidar data capture geometric features also observable in bird's-eye view overhead imagery, providing cues for localisation. Nevertheless, using satellite imagery for lidar localisation remains challenging, as overhead RGB images and range sensor scans are drastically different. Recent works were proposed to learn the metric localisation [57, 58] and place recognition [59] of radar and lidar against satellite imagery. Typically, learning place recognition requires *paired* data to form positive pairs in metric learning as was done in [59]. This, in turn, relies on accurate time-synced ground truth to query for a satellite image centred at the true centre position of each lidar scan, forming geometric one-to-one correspondences. In practice, as pure GPS measurements are usually not accurate enough for training high-performance place recognition pipelines, ground truth is created either through RTX/RTK information [19, 7] or by fusing GPS/INS data with Visual Odometry (VO) or full Simultaneous Localisation and Mapping (SLAM) systems [4, 29], requiring substantial cost and time. When the data in consideration are from the same modality, self-supervision with artificial paired data can be done by data augmentation for place recognition [44, 18] or representation learning [9]. However, self-supervised metric learning is largely unsolved between cross-modal data without known pairs.

We present a self-supervised method for learning lidar place recognition in overhead imagery from *unpaired* data. Suppose a lidar-equipped vehicle travels along a known route, we can, for example, sample satellite images uniformly along the route, forming a set of lidar and satellite data without geometric one-to-one correspondences. Our method then learns from unpaired data, effectively relaxing the need for an on-board GPS to collect time-synced, paired lidar and satellite data. Alternatively, if the known route has been previously surveyed by a vehicle with an on-board GPS, then position measurements collected from the prior traversal can also be used to query for satellite images along the route rather than sampling

uniformly. This effectively reduces the need to always equip an on-board GPS/INS when collecting training data. Figures 1 and 2 illustrate the difference between paired and unpaired data under the context of our problem set-up.

To learn from unpaired, cross-modal data, we first bridge the modality gap by utilising existing work on unpaired image-to-image translation. Our method then mines pseudo positive pairs with sequence alignment, where we propose a simple yet effective self-supervised learning strategy to de-alias the noisy affinity matrix resulting from modality difference. We demonstrate on public datasets that our method is on-par with supervised approaches trained with paired data when tested on unseen places.

II. Related Work

A. Learning Range Sensor Place Recognition

Learning approaches for range sensor place recognition can operate directly on point data or 2D scan images.

1) Point-based: Early work by Uy and Lee [60] utilises NetVLAD [2] after a PointNet [48] backbone to learn a global descriptor for retrieval. A common strategy utilised by recent methods seeks to learn per-point local descriptors first and then aggregate them into a global descriptor for place recognition using pooling [38, 31], normalisation [63], learned layers [38, 66, 14], or bag of words (BoW) [11].

2) Image-based: Sun et al. [56] expressed lidar data as bird's-eye view images and directly learns a global pose, which is used to seed a Monte Carlo Localiser for pose refinement. Saftescu et al. [49] expressed radar data as images in polar coordinates and learned rotation-invariant embeddings for place recognition via circular padding. Barnes and Posner [3] predicted keypoints and pixel-level local descriptors from radar images and aggregated them to a per-image global descriptor via pooling for place recognition. OverlapNet [8] uses 2D range, normal, intensity, and semantic images extracted from 3D lidar data to predict the overlap between scans as a proxy for detecting loop closure. OverlapNet was extended to a Transformer [61]-based architecture for rotation-invariant learning in [43], and to handle sequential data in [42].

B. Localisation with Overhead Imagery

Overhead imagery maps can be used to localise a forward-facing camera image in the geo-localisation problem. In this case, the localisation happens from different view perspectives (forward vs. overhead), but the nature of the query image and the map database are the same, i.e. RGB data. When a range sensor localises in overhead imagery maps, the view perspectives are consistent as range sensor scans are often expressed as bird's-eye view images. However, the data belong to different modalities, requiring alternative strategies for bridging the modality gap.

1) Cross-view, intra-modality: Early work by Lin et al. [34] learns cross-view geo-localisation with hand-crafted features and an SVM classifier. CVMNet [23] applies two streams of convolutional layers followed by a NetVLAD layer to learn cross-view matching with a weighted soft-margin ranking loss. Li et al. tackled cross-view retrieval by supplying orientation maps [35], predicting cross-view orientation [53], learning spatial attention [52], and using optimal feature transport [54].

2) Consistent view, cross-modality: The method in [12] accumulates lidar intensity sub-maps and directly localises against overhead imagery using Normalised Mutual Information. Some methods [13, 32] extract hand-crafted features from overhead imagery before comparing them against range sensor data. Zhu et al. [68] learned a matching probability between lidar grid-maps and satellite imagery to enhance a lidar SLAM pipeline. The work in [16] combines ground camera and lidar data to solve registration against overhead imagery in a correlation-maximisation approach. Tang et al. addressed the modality gap between range sensor data and satellite imagery by generating synthetic range sensor images [57, 58] or representing satellite imagery as points [59]. Prior learning-based methods, different from our work, have mostly relied on paired range sensor and satellite data for supervision.

Most recently, Kim et al. [30] trained a semantic segmentation network using hand-annotated satellite images, and extracted building outlines while also performing orthorectification. This practically converted satellite imagery to a representation similar to building outlines from OpenStreetMap (OSM). Localisation is then solved by computing mutual information between lidar scans against building outlines.

C. Localisation Using Other Publicly Available Resources

Other off-the-shelf resources, in particular OSM, have widely been applied for robot localisation. Concurrent works in [6, 17] match VO against road segments extracted from OSM for global localisation. The methods in [46, 64] bridge the modality gap between lidar and OSM by learning domain-invariant semantic descriptors. Cho et al. [10] designed a hand-crafted rotation-invariant descriptor based on the distance to buildings applicable to both lidar point-clouds and OSM data.

D. Cross-Modality Retrieval With Unpaired Data

Several recent works have targeted cross-modality retrieval with unpaired data. Yin et al. [65] applied a GAN-based style transfer that generates synthetic lidar images from radar images to achieve radar localisation in lidar maps. Jeong et al. [26] combined style transfer with joint feature space learning to match camera images with infra-red images. The method in [10] uses a hand-crafted descriptor for lidar and OSM data, and therefore does not require paired data for training, but is not directly applicable to our problem set-up. The method in [30] does not require paired lidar and satellite data, but relies on hand-annotated satellite imagery semantics to train semantic segmentation, which is another form of ground truth that is potentially time consuming to acquire.

Though promising, the method by Yin et al. [65] seeks to address the modality gap between two types of range sensors, while [26] targets RGB images and infra-red images, both having a much smaller modality gap than between lidar and overhead imagery. In our experiments, neither [65] nor [26] were sufficient in our problem of lidar place recognition in overhead imagery with unpaired data.

E. Unpaired Image-to-Image Translation

A core module of our pipeline involves generating synthetic lidar images from satellite images in an unpaired set-up. Unpaired image-to-image translation can be achieved using cycle-consistency [67] between the input and reconstructed images, or a shared latent space [36, 24]. These methods require learning bi-directional translation with two sets of generators and discriminators, while [47] achieves one-way unpaired image-to-image translation with contrastive learning. Several prior works were proposed to learn multi-modal image generation [24, 33, 1], but empirically they resulted in limited success in synthesising lidar images from overhead images in our experiments. Our choice fell on CycleGAN [67] as it is a well-established method.

III. Problem Overview

Suppose a vehicle travels along a known route and takes lidar scans at a certain frequency, resulting in a sequence of N lidar images $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$. Satellite images are queried along the route, forming a sequence of M satellite images $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_m\}$. In the simplest case, paired data are available through ground truth: for each lidar scan X_i, we have a satellite image Y_j sharing the same centre position, resulting in one-to-one correspondences with $N = M$.

Here, we consider a more general scenario where paired data are unavailable and the lidar and satellite sampling positions are not the same. For example, if the satellite images are sampled uniformly along the route. In this context, we want to find for each X_i the closest $Y_j \in \mathcal{Y}$, but, vitally, the correspondences are not known, and the mapping from $\mathcal{X}$ to $\mathcal{Y}$ is neither surjective nor injective.

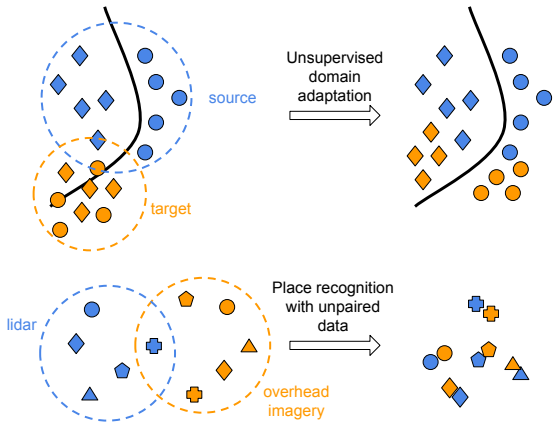


Fig. 3: **Top:** From unpaired data, UDA typically seeks to learn consistent decision boundaries between the source and target modalities. **Bottom:** In our problem, the goal is to find the nearest neighbours across modalities from continuously distributed data.

Here, we distinguish cross-modal place recognition with unpaired data from the perhaps better studied problem of unsupervised domain adaptation (UDA) [39, 50, 27]. Shown in Figure 3, UDA typically aims to learn consistent decision boundaries among source and target data. Our problem is different in that both lidar and overhead images are distributed *continuously* along a route, and neither can be appropriately classified into a discrete number of distinct categories. Our goal is instead to find the nearest neighbour from modality $\mathcal{Y}$ for each sample in $\mathcal{X}$, which can be achieved by sampling positive pairs from $\mathcal{X}$ and $\mathcal{Y}$ for metric learning. While this is trivial if $\mathcal{X}$ and $\mathcal{Y}$ are paired, our method seeks to target sampling positive pairs from unpaired data.

IV. Methodology

The core idea of our approach is to exploit the fact that, though without one-to-one correspondences, lidar and satellite data follow the same underlying sequence, as they are collected along the same known route. We begin in parallel by learning an embedding function for projecting a scan image to a vector-space descriptor using time consistency on lidar only (Section IV-B), and to generate synthetic lidar images from satellite imagery using CycleGAN [67] (Section IV-C). We can then use this projection to construct an affinity matrix by comparing synthetic lidar and real lidar scans.

Though the synthetic lidar images are realistic, there is no guarantee that they capture the regions of the scene as a real lidar situated at the centre of the satellite image. The affinity matrix will then be corrupted by significant signal aliasing. We propose a simple yet effective learned strategy to de-alias the affinity matrix and improve its signal-to-noise ratio, before performing sequence alignment (Section IV-D). Finally, pseudo pairs are selected from sequence alignment and used as positive pairs in metric learning to learn a final embedding-space projection for place recognition (Section IV-E).

A. Data Representation

In our problem set-up, we project 3D lidar data to the x-y plane to form bird's-eye view lidar images, where points with z value less than a threshold are discarded to remove ground points. The intensity in each pixel is the average intensity of all points projected to that pixel.



Fig. 4: A satellite image, a lidar image, and their polar counterparts.

Since the heading offset between lidar scans and satellite data is unknown, the place recognition pipeline must be rotation-invariant. Motivated by this, we convert lidar and satellite images to a polar coordinate representation, where the axes are range r and azimuth θ, and that a rotation in Cartesian space becomes a circular shift in the polar domain along the vertical axis. As CNNs are equivariant to vertical shifts, they can be easily trained to be rotation-invariant through data augmentation when applied to polar images. Examples of polar representations of satellite and lidar images are shown in Figure 4. While recent work proposes the Radon transform, which is $SE(2)$ equivariant, for lidar localisation [40], Radon

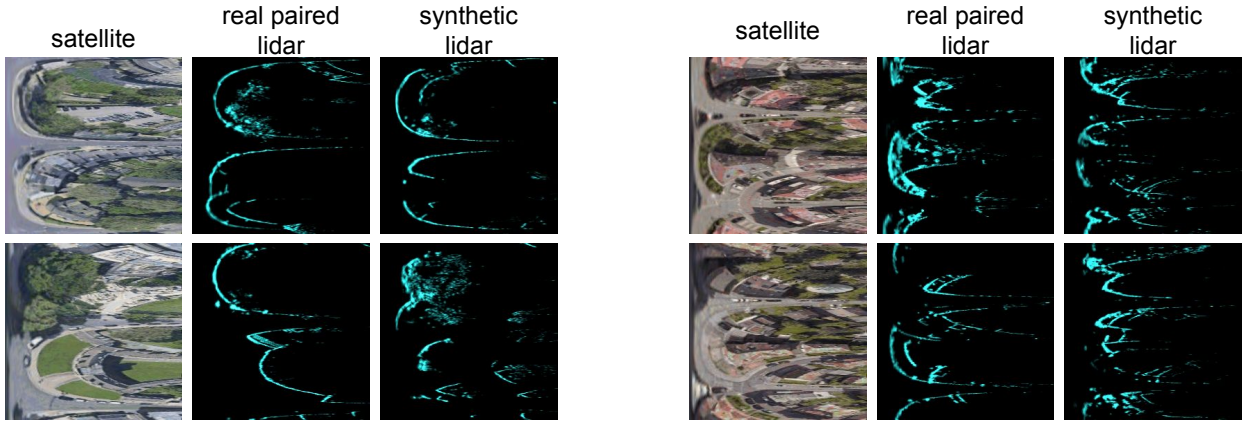


Fig. 5: Comparison between the input satellite image, an real lidar image scanned at the centre of the satellite image (not used during training, for visualisation only), and the synthetic lidar image generated from the satellite image, evaluated on the training set trajectories of RobotCar (left) and KITTI (right).

transform fails to preserve regions with a strong gradient, such as building edges, when applied to satellite images.

B. Descriptor Embedding for Lidar Images

We wish to learn a function F, parametrised by a neural network, that projects a lidar image to a descriptor space $\mathbb{R}^d$, where closeness in Euclidean space reflects closeness in Cartesian space geometrically. We train the function F in a self-supervised way using Siamese networks with a triplet margin loss. Specifically, given a polar lidar image X_i, we take its temporal neighbour taken τ frames earlier or later, $X_{i\pm\tau}$, to form a positive pair, and take a random sample X^- to form a negative pair, and minimise the following loss:

$$\mathcal{L}_{\text{emb}} = \Big[\|F(X_i) - F(X_{i\pm\tau})\| - \|F(X_i) - F(X^-)\| + m\Big]_+. \tag{1}$$

$[a]_+$ here denotes $\max(a, 0)$, and m is the triplet margin. In our experiments we set m to 1 and τ to 5. We apply random rotation augmentations to $X_i, X_{i\pm\tau}$, and X^- so that F learns to be rotation invariant.

C. Unpaired Satellite-to-Lidar Translation

To bridge the modality gap between satellite imagery and lidar, we use a variant of CycleGAN applied to the polar images. Specifically, we concatenate each polar lidar or satellite image with an additional single-channel image R, where the values of all pixels on column k of R is $\frac{k}{W}$, with W being the image width. Since height and width now denote azimuth and range values in polar representation, supplying R makes the generator and discriminator aware of the normalised range value of each pixel, and has shown to help stabilise training.

In CycleGAN, we seek to optimise generator functions $G_{X\to Y}, G_{Y\to X}$ and discriminator functions D_X, D_Y, parametrised by neural networks. $G_{X\to Y}$ maps an image from modality $\mathcal{X}$ to $\mathcal{Y}$, while $G_{Y\to X}$ is the counterpart. D_X and D_Y discriminate whether an image in modality $\mathcal{X}$ or $\mathcal{Y}$ is real or fake, respectively.

Given arbitrary X_i and Y_j, an adversarial loss can be applied to $G_{X\to Y}$ and D_Y :

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G_{X\to Y}, D_Y, X_i, Y_j, R) &= \mathbb{E}_{Y_j\sim\mathcal{Y}} \log D_Y(Y_j, R) \\ &+ \mathbb{E}_{X_i\sim\mathcal{X}} \log\Big(1 - D_Y\big(G_{X\to Y}\left(X_i, R\right), R\big)\Big),\end{aligned} \tag{2}$$

and a similar loss can be introduced for $G_{Y\to X}$ and D_X.

The cycle-consistency loss can be formulated as:

$$\begin{aligned}\mathcal{L}_{\text{cyc}} &= \mathbb{E}_{X_i\sim\mathcal{X}} \left\|X_i - G_{Y\to X}\big(G_{X\to Y}\left(X_i, R\right), R\big)\right\|_1 \\ &+ \mathbb{E}_{Y_j\sim\mathcal{Y}} \left\|Y_j - G_{X\to Y}\big(G_{Y\to X}\left(Y_j, R\right), R\big)\right\|_1.\end{aligned} \tag{3}$$

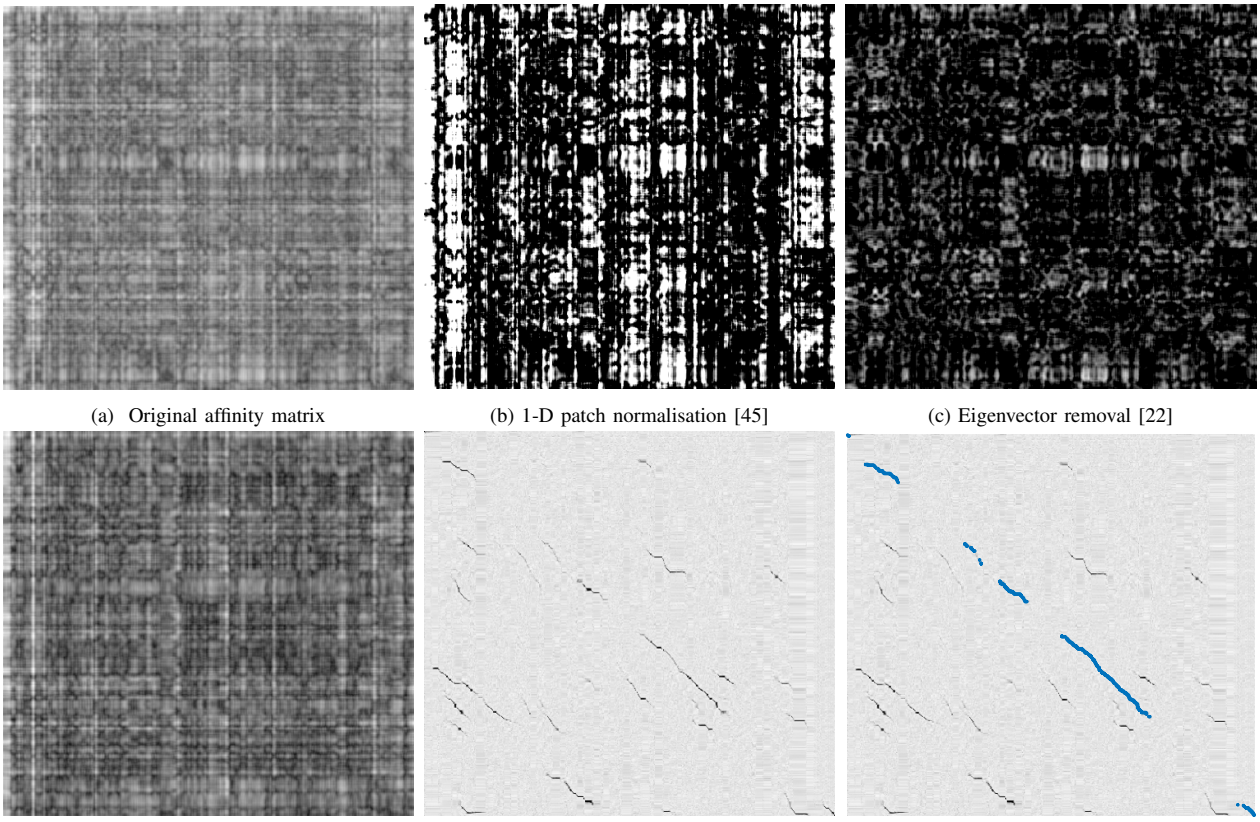
Finally, the full loss is:

$$\begin{aligned}\mathcal{L}_{\text{CycleGAN}} &= \mathcal{L}_{\text{GAN}}(G_{X\to Y}, D_Y, X_i, Y_j, R) \\ &+ \mathcal{L}_{\text{GAN}}(G_{Y\to X}, D_X, Y_j, X_i, R) + \lambda\mathcal{L}_{\text{cyc}},\end{aligned} \tag{4}$$

where λ was set to 100 in our experiments. The networks are optimised as:

$$G^*_{X\to Y}, G^*_{Y\to X} = \underset{G_{X\to Y}, G_{Y\to X}}{\operatorname{argmin}} \underset{D_X, D_Y}{\operatorname{argmax}} \mathcal{L}_{\text{CycleGAN}}. \tag{5}$$

As the majority of pixels in a lidar image are dark and only a small fraction has range returns, there is no guarantee synthetic lidar images will capture the same regions of the scene as an actual lidar situated at the centre of the satellite image. Figure 5 shows examples of synthetic lidar images compared to actual lidar images taken at the centre of the satellite image (not used in our experiments). Visibly, in many cases the synthetic lidar images appear significantly different than what an actual lidar would capture, which will result in finding false matches in the descriptor space. As such, relying on unpaired image-to-image translation only is insufficient, unlike the case of radar-to-lidar transfer [65] where locations with strong radar range return will likely also result in lidar return, and pixels with weak or no radar return will likely be not captured in a lidar scan.



(a) Original affinity matrix (b) 1-D patch normalisation [45] (c) Eigenvector removal [22]

(d) Dimensionality reduction [51, 37] (e) Ours (f) Pseudo pairs found

Fig. 6: (a) Affinity matrix computed from sequence `2011_10_03_0034` of the KITTI dataset, suffering from high signal aliasing with very poor local contrast. We show 4 methods for de-aliasing the affinity matrix, namely 1-D patch normalisation (b), eigenvector removal (c), dimensionality reduction (d), and our learned approach (e). (f) The highlighted pixels indicate pseudo pairs found from sequence alignment using a modified Smith-Waterman algorithm.

D. Sequence Alignment

Though individual nearest-neighbour matching with synthetic lidar images is inadequate, as shown in Section IV-C, pair-finding can be achieved with sequence alignment. Specifically, given a sequence of lidar images $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ and a sequence of satellite images $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_m\}$ queried along the same route, we first generate synthetic lidar images $\tilde{X}_j = G_{Y \to X}(Y_j)$ using the learned generator, forming a set of synthetic lidar images $\tilde{\mathcal{X}} = \{\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_m\}$. Next, we can construct an affinity matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$ where each element is the distance in descriptor space between X_i and $\tilde{X}_j$ after normalisation:

$$f_i = \frac{F(X_i) - \mu_X}{\sigma_X}, \quad \tilde{f}_j = \frac{F(\tilde{X}_j) - \mu_{\tilde{X}}}{\sigma_{\tilde{X}}} \qquad \mathbf{A}_{ij} = \left\| f_i - \tilde{f}_j \right\|, \tag{6}$$

where F is the learned embedding function from Section IV-B. $\mu_X \in \mathbb{R}^d$ and $\sigma_X \in \mathbb{R}^d$ are the mean and standard deviation of $\{F(X_1), \ldots, F(X_n)\}$, and similarly for $\mu_{\tilde{X}}$ and $\sigma_{\tilde{X}}$.

Since synthetic lidar images are not necessarily compatible with real lidar images as described in Section IV-C, the affinity matrix can be heavily corrupted by signal aliasing, resulting in a poor signal-to-noise ratio. Figure 6a shows the affinity matrix for a sequence from the KITTI dataset [19]. Various hand-crafted methods were proposed in prior work to enhance the contrast in the affinity matrix. SeqSLAM [45] proposes 1-D patch normalisation on the affinity matrix. Ho and Newman [22] performed eigendecomposition on the affinity matrix and reconstructed a rank-reduced one by removing eigenvectors corresponding to the largest eigenvalues. The methods in [51, 37] apply dimensionality reduction by keeping the descriptor's most descriptive $k \leq d$ dimensions. These methods were designed primarily for visual place recognition – where the mapping and localising sensors have the same sensing nature. In that case, the affinity matrix does not suffer from extreme signal aliasing as in our problem.

We design a learning-based approach for de-aliasing the affinity matrix trained on simulated data. First, we generate K random vectors $\{\xi_1, \xi_2, \ldots, \xi_k\}$, where each $\xi_i \in \mathbb{R}^d$ is sampled from a zero-mean, Gaussian distribution, and normalised to a unit sphere. Next, we form a random dynamic sequence of length P, $\mathbf{\Phi} = \{\phi_1, \ldots, \phi_p\}$, by travelling from ξ_1 to ξ_k, taking step sizes of $0, 1$, or 2 with various probabilities each. This procedure is repeated, forming a different random dynamic sequence of length Q, $\mathbf{\Psi} = \{\psi_1, \ldots, \psi_q\}$. We can construct an affinity matrix $\mathbf{A}_s \in \mathbb{R}^{P \times Q}$ between $\mathbf{\Phi}$ and $\mathbf{\Psi}$. Here, P and Q can either be less than, equal to, or larger than K. The probability for choosing each step size is a design choice. Empirically, they do not seem to greatly impact

performance.

We then add artificial signal aliasing to corrupt the simulated affinity matrix. We take inspiration from [22], yet, instead of removing eigenvectors, we introduce signal aliasing by *adding* an eigenvector from the affinity matrix of a KITTI sequence whose real and synthetic lidar embeddings result in heavy aliasing. The detailed steps of the procedure are found in Algorithm 1 and a visual example of a simulated affinity matrix A_s and its alias-corrupted version $\mathbf{A}'_s$ is shown in Figure 7.

Algorithm 1: Adding Signal Aliasing

Input:
$\mathbf{A}_s \in \mathbb{R}^{P\times Q}$ # affinity matrix from simulated data
$\{f_1, \ldots, f_n\}$, $\{\tilde{f}_1, \ldots, \tilde{f}_m\}$
Output:
$\mathbf{A}'_s$ # affinity matrix with added signal aliasing
Procedure:
$\mathbf{f} \in \mathbb{R}^{d\times(N+M)} \leftarrow \begin{bmatrix} f_1 & \ldots & f_n & \tilde{f}_1 & \ldots & \tilde{f}_m \end{bmatrix}$
$\mathbf{W} \in \mathbb{R}^{(N+M)\times(N+M)} \leftarrow$ initialise
for $i = 1, 2, \cdots, N+M$ **do**
  for $j = 1, 2, \cdots, N+M$ **do**
    # $\mathbf{f}_i$ and $\mathbf{f}_j$ are the ith and j^{th} columns of $\mathbf{f}$
    $\mathbf{W}_{ij} = \|\mathbf{f_i} - \mathbf{f}_j\|$
$\mathbf{\Lambda}, \mathbf{V} \leftarrow \text{eigendecomposition}(\mathbf{W})$
$\lambda^*, \mathbf{v}^* \leftarrow$ largest eigenvalue and the corresponding eigenvector
$\mathbf{v}^* \leftarrow \text{randompermute}(\mathbf{v}^*)$
$\mathbf{N} \in \mathbb{R}^{(N+M)\times(N+M)} \leftarrow \mathbf{v}^{*\mathrm{T}}\lambda^*\mathbf{v}^*$
random crop to a $P \times Q$ patch
$\mathbf{N} \in \mathbb{R}^{P\times Q} \leftarrow \text{randomcrop}(\mathbf{N}, P, Q)$
$\mathbf{A}'_s \leftarrow \mathbf{A}_s + \mathbf{N}$
$\mathbf{A}'_s \leftarrow \mathbf{A}'_s / \max(\mathbf{A}'_s)$

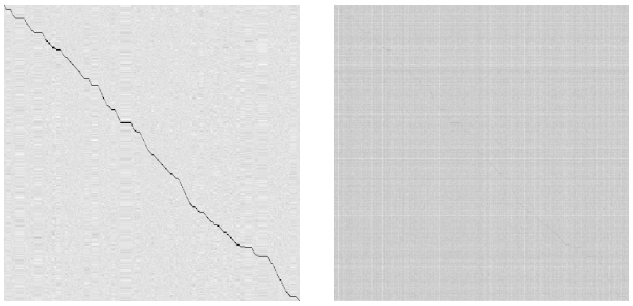


Fig. 7: Example of a simulated affinity matrix $\mathbf{A}_s$ (**left**) and its aliasing-corrupted version $\mathbf{A}'_s$ (**right**). We train a neural network that takes $\mathbf{A}'_s$ as input and recovers the original, high-contrast affinity matrix $\mathbf{A}_s$.

We can treat $\mathbf{A}_s$ and $\mathbf{A}'_s$ as single-channel images and apply Pix2Pix [25] to learn to recover a clean affinity matrix from its alias-corrupted counterpart:

$$\mathbf{A}_s = G_{DA}(\mathbf{A}'_s), \tag{7}$$

where $G_{DA} : \mathbb{R}^{H\times W} \rightarrow \mathbb{R}^{H\times W}$ is a matrix de-aliasing function parametrised by a neural network. After G_{DA} is optimised, we apply G_{DA} to the aliased affinity matrix $\mathbf{A}$ for de-aliasing. Finally, we use the modified Smith-Waterman algorithm in [22] on the de-aliased affinity matrix for sequence alignment, where pseudo pairs are found forming a set of pairs $\mathcal{S}$, with each $(X_p, Y_q) \in \mathcal{S}$ being an aligned pair found by sequence alignment. Figure 6 compares the effect of existing hand-crafted approaches and our novel learning-based method on de-aliasing the affinity matrix.

E. Learning Place Recognition from Pseudo Pairs

After pseudo pairs are selected from sequence alignment, we use them as positive pairs in metric learning for place recognition. Formally, we wish to learn embedding functions F_X and F_Y that project images of modality $\mathcal{X}$ and $\mathcal{Y}$ respectively to $\mathbb{R}^D$. We parameterise F_X and F_Y by neural networks. Here F_X and F_Y are separate and different from F as in Section IV-B, which was used for constructing the affinity matrix. To optimise F_X and F_Y, we aim to minimise a bi-directional triplet margin loss:

$$\mathcal{L}_{\text{PR}} = \Big[\, \|F_X(X_p) - F_Y(Y_q)\| - \big\|F_X(X_p) - F_Y(Y^-)\big\| + m \Big]_+ + \Big[\, \|F_Y(Y_q) - F_X(X_p)\| - \big\|F_Y(Y_q) - F_X(X^-)\big\| + m \Big]_+, \tag{8}$$

where $(X_p, Y_q) \in \mathcal{S}$, and the negative samples X^- and Y^- are random samples from $\mathcal{X}$ and $\mathcal{Y}$. Here we set the triplet margin m to 0.1. We again apply random rotation augmentations to the images so F_X and F_Y learn to be rotation-invariant.

F. Network Architecture and Implementation Details

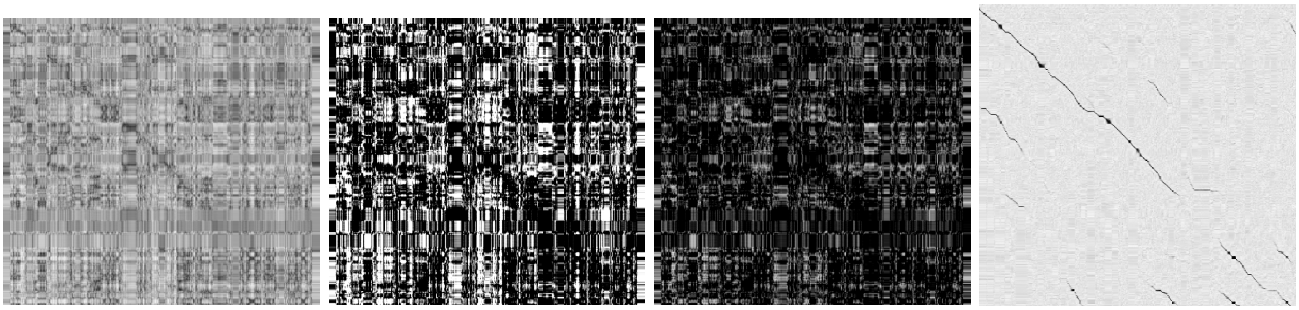
For the image translation networks $G_{x\rightarrow Y}, G_{Y\rightarrow X}$, and G_{DA}, we use the ResNet [21] generator from the authors' official implementation [1]. For learning descriptor embeddings, we use the convolution layers of a VGG16 [55] backbone followed by a NetVLAD layer [2] for F, and the convolution layers of a ResNet18 backbone, followed by a NetVLAD layer for F_X and F_Y. On lidar and synthetic lidar images, we add a single convolution layer at the top before VGG16 or ResNet18 to convert the single-channel image to 3 channels. We set the embedding dimensions as $d = 256$ and $D = 2048$.

All of our training is conducted in PyTorch. We use RMSProp as the optimiser with a fixed learning rate of 0.0001 in CycleGAN training and ADAM with a fixed learning rate of 0.0002 in all other modules. As no ground truth data is used, we cannot split the training data to form a validation set. Instead, we arbitrarily terminate training when the training loss has stabilised for 10 epochs.

V. Experimental Results

Our method is validated on the Oxford Radar RobotCar Dataset [4], which uses two Velodyne HDL-32E lidars mounted in a tilted configuration where we take just the left lidar, and the KITTI Dataset (raw data), [19], which uses a Velodyne HDL-64E lidar mounted on-top. Unlike existing

[1]https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix



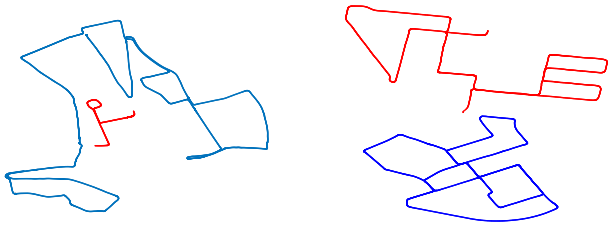
(a) Original affinity matrix (b) 1-D patch normalisation [45] (c) Eigenvector removal [22] (d) Ours

Fig. 8: Results of applying our learned method compared to hand-crafted techniques on the affinity matrix of sequence no.2 from RobotCar.

methods that typically solve place recognition or loop closure when revisiting a place, in our experiments the trajectories at inference time are completely unseen during training.

As RobotCar features repeated traversals of the same route, we split the trajectory into training and test sets with no overlap, as shown in Figure 9a. We use lidar data collected along the training trajectories from sequences no. 2 and 5 and query satellite data according to the GPS-INS timestamps of sequence no. 6.

On KITTI, the training set consists of lidar data from two long residential sequences, `2011_10_03_0027` and `2011_10_03_0034`. To simulate using GPS data from a prior survey, we take the paired GPS data but uniformly divide each sequence into 5000 segments based on distance and find the centre of each segment. We then sample satellite images at these centres with an added random error of $-2\,\mathrm{m}$ to $2\,\mathrm{m}$ to simulate the survey vehicle following a slightly different route. The test set consists of another long residential sequence, `2011_09_30_0028`, where we discard the first 1000 frames to avoid overlap with training data, shown in Figure 9b.



(a) RobotCar (b) KITTI

Fig. 9: The RobotCar data were split into training (blue) and test (red). For KITTI, the test set is `2011_09_30_0028` with the first 1000 frames removed (red), as they overlap with `2011_10_03_0027` (blue).

We consider single-frame localisations only and do not use sequential information. We compare each lidar frame against a map database of all satellite images in the descriptor space using the learned networks F_X and F_Y.

1) Retrieval Accuracy: All test set satellite images $\{Y_j\}$ are mapped to the descriptor space, forming $\{F_Y(Y_j)\}$ where $F_Y(Y_j) \in \mathbb{R}^D \; \forall j$. For each lidar image X_i at test time, we map it to descriptor space as $F_X(X_i) \in \mathbb{R}^D$, and calculate the Euclidean distances against all satellite data descriptors. For each lidar image, we find the top-1 match and the top 1% matches with the smallest descriptor distance. We evaluate the percentage of retrievals that fall within a certain threshold of the lidar's true position.

As learning lidar-only place recognition in satellite imagery maps is relatively unstudied, we compare against a supervised approach using the same metric learning approach and network choices as in Section IV-E, except with paired data. As ablation, we further compare our method for unpaired data with a baseline method that directly uses the descriptor embedding function F trained on real lidar images only as in Section IV-B, and directly operate on synthetic lidar images generated from satellite images using the approach in Section IV-C. Specifically, for the unpaired baseline, all test set satellite images are mapped to descriptor space as $\{F\big(G_{Y\to X}(Y_j)\big)\}$, and each lidar image at test time is mapped to descriptor space as $F(X_i)$. The unpaired baseline is similar in spirit as [65] but applied to lidar-satellite place recognition and makes no attempt at sequence alignment.

2) Precision and Recall: For this evaluation, we sample various descriptor distance thresholds from the minimum distance between any $F_X(X_i)$ to any $F_Y(Y_j)$ to the maximum, and consider matches less than the threshold positives and the rest negatives. Two locations along the test set trajectory are considered true matches if their distance is less than $50\,\mathrm{m}$ and false if their distance is more than $75\,\mathrm{m}$. The precision and recall for all descriptor-distance thresholds are computed to generate the precision-recall curves.

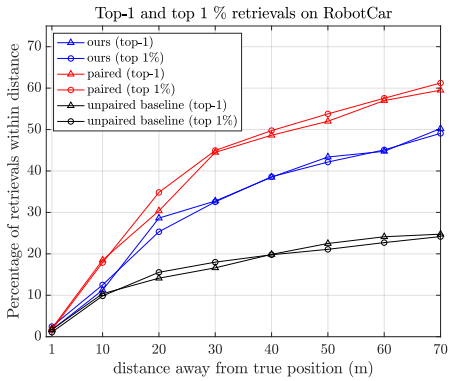


Fig. 10: Retrieval performance on the test set of RobotCar.

A. RobotCar Dataset

The affinity matrix for sequence no.2 of RobotCar is shown in Figure 8a. Here the affinity matrix has much less signal

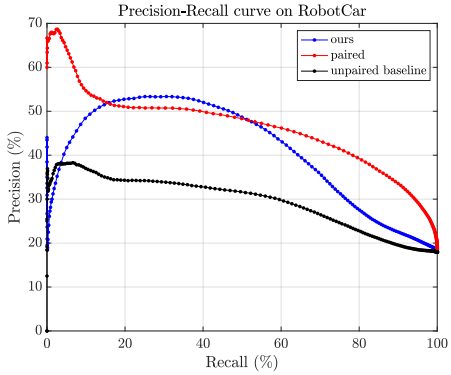


Fig. 11: Precision-recall curve on the test set of RobotCar.

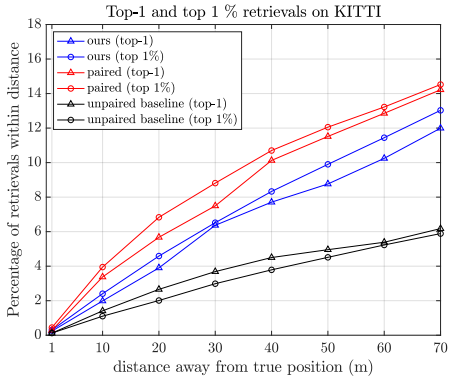


Fig. 12: Retrieval performance on the test set of KITTI.

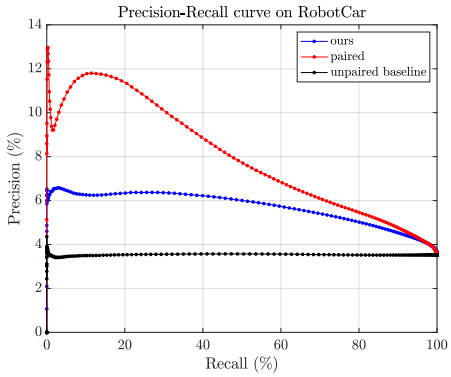


Fig. 13: Precision-recall curve on the test set of KITTI.

aliasing than the KITTI dataset (Figure 6a). Though hand-crafted techniques such as 1-D patch normalisation and eigenvector can further increase local contrast to a sufficient extent, we show qualitatively in Figure 8 that our learned method is much better at reducing aliasing.

The test set trajectory of the RobotCar dataset features approximately $1\,\mathrm{km}$ of urban environment, with 800 lidar scans in total. The retrieval performances of our method compared to the supervised method and the unpaired baseline are shown in Figure 10. Though slightly under-performing than the supervised method, more than half of top-1 retrievals using our unpaired method can localise correctly within $70\mathrm{m}$ of the true position on the RobotCar dataset. Figure 11 shows the precision-recall curve, where our method has higher precision than using paired data for recall between 20% and 50%.

B. KITTI Dataset

KITTI features much more challenging data than RobotCar for lidar place recognition using overhead imagery. Firstly, KITTI uses a Velodyne HDL-64E lidar compared to Velodyne HDL-32E. Though the higher vertical resolution is ideal for many lidar-based applications, it increases complexity in the resulting lidar images, making it less likely for the synthetic lidar images to be compatible with real lidar scans. This has made it extremely challenging to perform sequence alignment from unpaired data, as indicated by the significantly higher levels of aliasing (Figure 6a) than RobotCar (Figure 8a). Moreover, the test set is in a residential area with many similar places and fewer distinct landmarks, resulting in a high false positive rate. Finally, our test set of KITTI spans a total distance of approximately $4\,\mathrm{km}$, having a much larger satellite map database, naturally making retrieval difficult.

The retrieval performance and precision-recall curve are shown in Figures 12 and 13. It can be seen that even the fully-supervised baseline struggles in this environment. Nevertheless, our method significantly outperforms the unpaired baseline method.

C. Unpaired Radar Place Recognition in Overhead Imagery

Our method was originally designed for lidar data but can be applied to radar data through radar-to-lidar image translation. Specifically, on the RobotCar dataset which also has on-board Navtech radar data, given radar images $\mathcal{Z} = \{Z_1, Z_2, \dots\}$, we apply CycleGAN [67] to learn a function $G_{Z\to X} : \mathbb{R}^{H\times W} \to \mathbb{R}^{H\times W}$ that maps a polar radar image to its synthetic lidar counterpart. Figure 14 shows examples of radar images and the corresponding synthetic lidar images after style transfer.

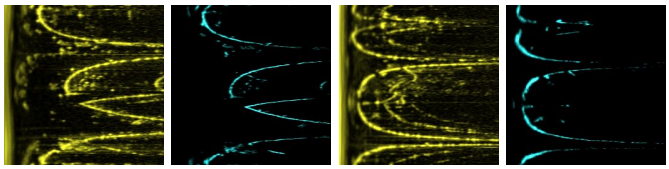


Fig. 14: Radar images (in orange) and corresponding synthetic lidar images.

Then, without further re-training, we utilise networks learned with lidar data from our unpaired approach and directly apply them to synthetic lidar images from radar input. Specifically, taking F_X, F_Y trained for lidar-satellite place recognition with found pseudo pairs, at test time, each radar image Z_i is mapped to descriptor space as $F_X\big(G_{Z\to X}(Z_i)\big)$, and compared against the map database of satellite image descriptors $\{F_Y(Y_j)\}$. We compare our approach against a supervised method trained on paired radar and satellite images, using the metric learning approach in Section IV-E. The retrieval performance and precision-recall curve are shown in Figures 15 and 16. Notably, the unpaired approach has higher retrieval accuracy than the supervised, paired approach on smaller error thresholds.

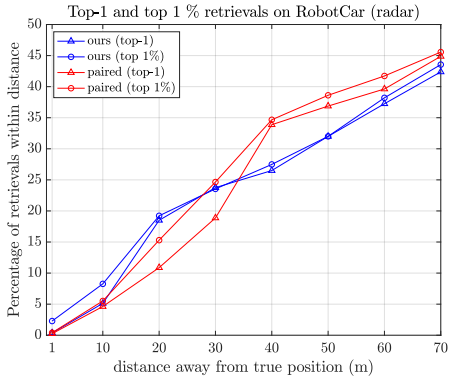


Fig. 15: Retrieval performance on the test set of RobotCar with radar data.

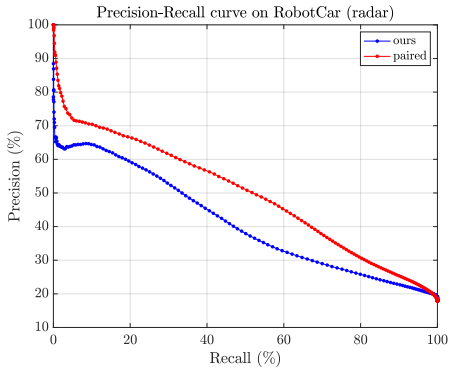


Fig. 16: Precision-recall curve on the test set of RobotCar with radar data.

VI. Conclusion and Future Work

In this paper, we show how lidar place recognition in a map of satellite images can be solved without any paired data. Our approach relaxes the need for an on-board, time-synchronised GPS/INS when collecting on-road lidar data for high-precision pose ground truth, as long as the route taken is known from a previous survey or traversal. Though the performance of place recognition in overhead imagery is far from lidar-to-lidar localisation, this capability nevertheless allows a mobile robot to travel to an unvisited place and still localise to a certain extent, and can also add an extra layer of redundancy in standard navigation applications.

Here we focus on solving the pairing problem from unpaired data and rely on metric learning for solving place recognition. Using data with global pose ground truth, future work can target how to further bridge the domain gap between lidar data and overhead imagery to enhance the localisation quality.

References

[1] Amjad Almahairi, Sai Rajeshwar, Alessandro Sordoni, Philip Bachman, and Aaron Courville. Augmented Cyclegan: Learning Many-to-Many Mappings from Unpaired Data. In *International Conference on Machine Learning*, pages 195–204. PMLR, 2018.

[2] 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.

[3] Dan Barnes and Ingmar Posner. Under the Radar: Learning to Predict Robust Keypoints for Odometry Estimation and Metric Localisation in Radar. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Paris, 2020. URL https://arxiv.org/abs/2001.10789.

[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] 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.

[6] 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.

[7] Keenan Burnett, David J Yoon, Yuchen Wu, Andrew Zou Li, Haowei Zhang, Shichen Lu, Jingxing Qian, Wei-Kang Tseng, Andrew Lambert, Keith YK Leung, et al. Boreas: A multi-season autonomous driving dataset. *arXiv preprint arXiv:2203.10168*, 2022.

[8] Xieyuanli Chen, Thomas Läbe, Andres Milioto, Timo Röhling, Olga Vysotska, Alexandre Haag, Jens Behley, and Cyrill Stachniss. OverlapNet: Loop Closing for LiDAR-based SLAM. In *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, July 2020. doi: 10.15607/RSS.2020.XVI.009.

[9] Chen, Ting and Kornblith, Simon and Norouzi, Mohammad and Hinton, Geoffrey. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[10] Younghun Cho, Giseop Kim, Sangmin Lee, and Jee-Hwan Ryu. OpenStreetMap-based LiDAR Global Localization in Urban Environment without a Prior LiDAR Map. *IEEE Robotics and Automation Letters*, 7(2):4999–5006, 2022.

[11] Yunge Cui, Xieyuanli Chen, Yinlong Zhang, Jiahua Dong, Qingxiao Wu, and Feng Zhu. BoW3D: Bag

of Words for Real-Time Loop Closing in 3D LiDAR SLAM. *IEEE Robotics and Automation Letters*, 2022.

[12] 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.

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

[14] 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.

[15] 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.

[16] Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens, and Rainer Stiefelhagen. Continuous Self-Localization on Aerial Images Using Visual and Lidar Sensors. *arXiv preprint arXiv:2203.03334*, 2022.

[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] Matthew Gadd, Daniele De Martini, and Paul Newman. Contrastive Learning for Unsupervised Radar Place Recognition. In *2021 20th International Conference on Advanced Robotics (ICAR)*, pages 344–349, 2021. doi: 10.1109/ICAR53236.2021.9659335.

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

[20] 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.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Kin Leong Ho and Paul Newman. Detecting Loop Closure with Scene Sequences. *International journal of computer vision*, 74(3):261–286, 2007.

[23] Sixing Hu, Mengdan Feng, Rang MH Nguyen, and Gim Hee Lee. CVM-Net: Cross-View Matching Network for Image-Based Ground-to-Aerial Geo-Localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7258–7267, 2018.

[24] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal Unsupervised Image-to-Image Translation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 172–189, 2018.

[25] 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.

[26] Somi Jeong, Seungryong Kim, Kihong Park, and Kwanghoon Sohn. Learning to Find Unpaired Cross-Spectral Correspondences. *IEEE Transactions on Image Processing*, 28(11):5394–5406, 2019.

[27] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum Class Confusion for Versatile Domain Adaptation. In *European Conference on Computer Vision*, pages 464–480. Springer, 2020.

[28] 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.

[29] Giseop Kim, Yeong Sang Park, Younghun Cho, Jinyong Jeong, and Ayoung Kim. MulRan: Multimodal Range Dataset for Urban Place Recognition. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. Submitted. Under Review.

[30] Jonghwi Kim, Yonghoon Cho, and Jinwhan Kim. Urban Localization Based on Aerial Imagery by Correcting Projection Distortion. *Autonomous Robots*, pages 1–14, 2022.

[31] Jacek Komorowski. Minkloc3d: Point Cloud Based Large-Scale Place Recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1790–1799, 2021.

[32] 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.

[33] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse Image-to-Image Translation via Disentangled Representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 35–51, 2018.

[34] Tsung-Yi Lin, Serge Belongie, and James Hays. Cross-View Image Geolocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 891–898, 2013.

[35] Liu Liu and Hongdong Li. Lending Orientation to Neural Networks for Cross-View Geo-Localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5624–5633, 2019.

[36] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised Image-to-Image Translation Networks. In *Advances in Neural Information Processing Systems*,

pages 700–708, 2017.

[37] Yang Liu and Hong Zhang. Visual Loop Closure Detection with a Compact Image Descriptor. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1051–1056. IEEE, 2012.

[38] 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.

[39] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep Transfer Learning with Joint Adaptation Networks. In *International conference on machine learning*, pages 2208–2217. PMLR, 2017.

[40] Sha Lu, Xuecheng Xu, Li Tang, Rong Xiong, and Yue Wang. DeepRING: Learning Roto-Translation Invariant Representation for LiDAR based Place Recognition. *arXiv preprint arXiv:2210.11029*, 2022.

[41] Lun Luo, Si-Yuan Cao, Bin Han, Hui-Liang Shen, and Junwei Li. BVMatch: LiDAR-Based Place Recognition Using Bird's-Eye View Images. *IEEE Robotics and Automation Letters*, 6(3):6076–6083, 2021.

[42] Junyi Ma, Xieyuanli Chen, Jingyi Xu, and Guangming Xiong. SeqOT: A Spatial-Temporal Transformer Network for Place Recognition Using Sequential LiDAR Data. *arXiv preprint arXiv:2209.07951*, 2022.

[43] Junyi Ma, Jun Zhang, Jintao Xu, Rui Ai, Weihao Gu, and Xieyuanli Chen. OverlapTransformer: An Efficient and Yaw-Angle-Invariant Transformer Network for LiDAR-Based Place Recognition. *IEEE Robotics and Automation Letters*, 2022.

[44] Nate Merrill and Guoquan Huang. Lightweight Unsupervised Deep Loop Closure. 2018.

[45] Michael J Milford and Gordon F Wyeth. SeqSLAM: Visual Route-Based Navigation for Sunny Summer Days and Stormy Winter Nights. In *2012 IEEE international conference on robotics and automation*, pages 1643–1649. IEEE, 2012.

[46] 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.

[47] Taesung Park, Alexei A Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive Learning for Unpaired Image-to-Image Translation. In *European conference on computer vision*, pages 319–345. Springer, 2020.

[48] 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.

[49] Stefan Saftescu, Matthew Gadd, Daniele De Martini, Dan Barnes, and Paul Newman. Kidnapped Radar: Topological Radar Localisation using Rotationally-Invariant Metric Learning. *arXiv preprint arXiv: 2001.09348*, 2020.

[50] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum Classifier Discrepancy for Unsupervised Domain Adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3723–3732, 2018.

[51] Stefan Schubert, Peer Neubert, and Peter Protzel. Unsupervised Learning Methods for Visual Place Recognition in Discretely and Continuously Changing Environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4372–4378. IEEE, 2020.

[52] Yujiao Shi, Liu Liu, Xin Yu, and Hongdong Li. Spatial-Aware Feature Aggregation for Image Based Cross-View Geo-Localization. *Advances in Neural Information Processing Systems*, 32, 2019.

[53] Yujiao Shi, Xin Yu, Dylan Campbell, and Hongdong Li. Where Am I Looking At? Joint Location and Orientation Estimation by Cross-View Matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4064–4072, 2020.

[54] Yujiao Shi, Xin Yu, Liu Liu, Tong Zhang, and Hongdong Li. Optimal Feature Transport for Cross-View Image Geo-:ocalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11990–11997, 2020.

[55] 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.

[56] 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.

[57] 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.

[58] 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.

[59] Tim Y. Tang, Daniele De Martini, and Paul Newman. Get to the Point: Learning Lidar Place Recognition and Metric Localisation Using Overhead Imagery. In *Proceedings of Robotics: Science and Systems*, Virtual, July 2021. doi: 10.15607/RSS.2021.XVII.003.

[60] 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.

[61] 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.

[62] Kavisha Vidanapathirana, Peyman Moghadam, Ben Harwood, Muming Zhao, Sridha Sridharan, and Clinton Fookes. Locus: Lidar-Based Place Recognition Using Spatiotemporal Higher-Order Pooling. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5075–5081. IEEE, 2021.

[63] Kavisha Vidanapathirana, Milad Ramezani, Peyman Moghadam, Sridha Sridharan, and Clinton Fookes. LoGG3D-Net: Locally Guided Global Descriptor Learning for 3D place Recognition. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2215–2221. IEEE, 2022.

[64] 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.

[65] Huan Yin, Yue Wang, Jun Wu, and Rong Xiong. Radar Style Transfer for Metric Robot Localisation on Lidar Maps. *CAAI Transactions on Intelligence Technology*, 2022.

[66] 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.

[67] 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.

[68] Minzhao Zhu, Yi Yang, Wenjie Song, Meiling Wang, and Mengyin Fu. AGCV-LOAM: Air-Ground Cross-View Based LiDAR Odometry and Mapping. In *2020 Chinese Control And Decision Conference (CCDC)*, pages 5261–5266. IEEE, 2020.

7
Concluding Remarks

This thesis investigated how satellite imagery can be utilised to localise a robot equipped with ground range sensors, in the absence of GPS, prior sensor map, or other data sources. Our methods are learned end-to-end, allowing neural networks to discover any mutual information between range sensor data and overhead imagery, without relying on any hand-crafted features or techniques designed for a specific type of environment. In the opening section in Chapter 1 we remarked that the use of overhead imagery as a prior source of shape and appearance for localising with range sensors is both a challenging and under explored combination. With this in mind, motivated by the upsides described in Chapter 1 and the research gaps pointed out in Chapter 2, this thesis has made the contributions set out in Chapters 3 to 6.

Specifically, Chapters 3 and 4 tackled the modality difference by generating synthetic range sensor images using paired satellite and range sensor data as inputs. The synthetic range sensor image is pixel-wise aligned with the input satellite image, but shares the same appearance as the input range sensor image, and can be registered against it to solve the metric localisation. The method presented in Chapter 5 learns a grey-scale occupancy image from overhead imagery, which is then used to transform the overhead imagery into a point-cloud. The resulting point-cloud is then compared against points from a lidar or radar scan for point-cloud alignment or retrieval. Finally, Chapter 6 shows how to discover

pairs from two different and unsynchronised sequences of unpaired range sensor and overhead imagery data using sequence alignment, allowing to learn the cross-modality retrieval without paired data.

However, further research work could be done and we conclude the thesis with some forward thoughts on where this research direction could flow in the future:

1. As a proof of concept and an effort to demonstrate a standalone methodology using overhead imagery only, the work in this thesis attempted to solve localisation with the minimal information possible. We made no use of pre-trained networks for satellite imagery semantic segmentation or edge detection, ground camera data, trajectory estimates from odometry or SLAM, or other public data such as OSM. However, in real-world autonomous navigation, the preferred approach is often to use the maximal information possible for redundancy. Therefore, an interesting future work is to implement techniques presented in this thesis within a larger localisation pipeline that also makes use of, for example, SLAM and loop closures, other ways to extract geometric information from overhead imagery, and possibly methods to localise range sensors in OSM.

2. Satellite images suffer from projection distortion errors, which are prominent especially in taller buildings. Correcting for projection distortion in overhead imagery, also known as the process of orthorectification, could improve localisation quality. While prior methods mostly use hand-designed methods for orthorectification, one can apply machine learning to correct for projection distortion through a data-driven approach.

3. Recently, public 3D imagery is made available from providers such as Google Earth [1]. Once the 3D surface meshes of a place are extracted, then they can be used as a powerful and versatile map source for localisation, possibly in conjunction with satellite imagery.

[1] `https://earth.google.com/`

4. Semantic segmentation can be applied to satellite images to help bridge the modality gap against range sensor data, for example by extracting the contours of buildings. For self-supervised localisation, future work can attempt to apply unsupervised semantic segmentation on satellite imagery to avoid the need to collect hand-labelled semantic ground truth.

Appendices

A

Preparing Overhead Imagery Data

Given a known latitude and longitude, satellite imagery can be queried off the internet using Google Maps[1] or Bing Maps[2] API.

As the resolution in m/pixel of the satellite images is known, they are processed to have the same resolution as range sensor images, leaving the image registration problem to be in $SE(2)$. Specifically, under the assumption that the Earth is a sphere, after supplying a "zoom level" argument, the resolution of satellite images in m/pixel can be calculated as[3]:

$$\text{resolution} = 156543.03392 \frac{\cos(lat)}{2^z}, \tag{A.1}$$

where lat is the latitude in degrees, and z is the zoom level. In this thesis we have used zoom levels 17 or 18, depending on the sensing range of the lidar or radar.

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

[2] https://docs.microsoft.com/en-us/bingmaps/

[3] https://groups.google.com/g/google-maps-js-api-v3/c/hDRO4oHVSeM/m/osOYQYXg2oUJ

B

Self-Supervised Localisation between Range Sensors and Overhead Imagery

This article was published as a conference proceeding at Robotics: Science and System XVI in July 2020.

Robotics: Science and Systems 2020
Corvalis, Oregon, USA, July 12-16, 2020

Self-Supervised Localisation between Range Sensors and Overhead Imagery

Tim Y. Tang*, Daniele De Martini*, Shangzhe Wu†, and Paul Newman*
*Mobile Robotics Group, University of Oxford †Visual Geometry Group, University of Oxford
{ttang, daniele, szwu, pnewman}@robots.ox.ac.uk

***Abstract*—Publicly available satellite imagery can be an ubiquitous, cheap, and powerful tool for vehicle localisation when a prior sensor map is unavailable. However, satellite images are not directly comparable to data from ground range sensors because of their starkly different modalities. We present a learned metric localisation method that not only handles the modality difference, but is cheap to train, learning in a self-supervised fashion without metrically accurate ground truth. By evaluating across multiple real-world datasets, we demonstrate the robustness and versatility of our method for various sensor configurations. We pay particular attention to the use of millimetre wave radar, which, owing to its complex interaction with the scene and its immunity to weather and lighting, makes for a compelling and valuable use case.**

I. Introduction

The ability to localise relative to an operating environment is central to robot autonomy. Localisation using range sensors, such as lidars [25, 44] and, more recently, scanning millimetre wave radars [33, 36], is an established proposition. Both are immune to changing lighting conditions and directly measure scale, while the latter adds resilience to weather conditions.

Current approaches to robot localisation typically rely on a prior map built using a sensor configuration that will also be equipped on-board, for example a laser map for laser localisation. This paper looks at an alternative method. Public overhead imagery such as satellite images can be a reliable map source, as they are readily available, and often captures information also observable, albeit perhaps in some complex and incomplete way, by sensors on the ground. We can pose the localisation problem in a natural way: find the pixel location of a sensor in an overhead (satellite) image given range data taken from the ground. The task is, however, non-trivial because of the drastic modality difference between satellite image and sparse, ground-based radar or lidar.

Recent work on learning to localise a ground scanning radar against satellite images [39] provides a promising direction which addresses the modality difference by first generating a synthetic radar image from a satellite image. The synthetic image is then "compared" against a live radar image for pose estimation. Such an approach learns metric, cross-modality localisation in an end-to-end fashion, and therefore does not require hand-crafted features limited to a specific environment.

The method in [39] trains a multi-stage network, and needs pixel-wise aligned radar and satellite image pairs for supervision at all stages. This in turn requires sub-metre accurate ground truth position and sub-degree accurate ground truth heading. In practice, collecting accurate ground truth requires high-end GPS/Inertial Navigation System (INS), and possibly bundle adjustment along with other on-board sensor solutions, bringing in burdens in terms of cost and time consumption.

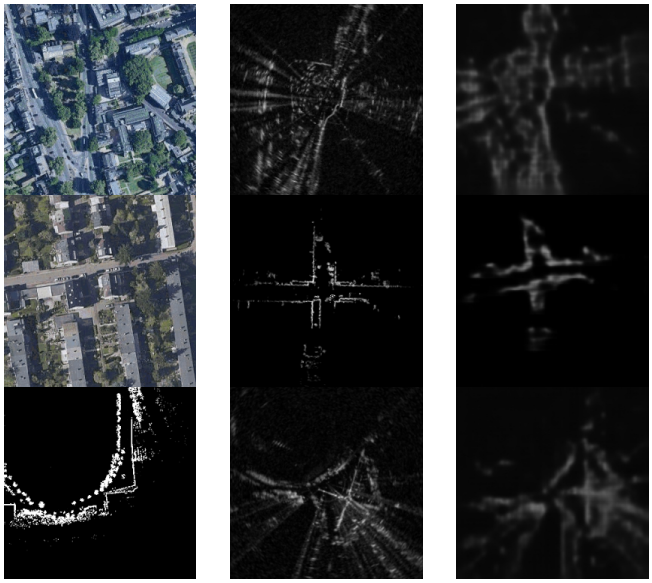


Fig. 1: Given a map image of modality $\mathcal{A}$ (left) and a live data image of modality $\mathcal{B}$ (middle), we wish to find the unknown $SE(2)$ offset between them. To do so, our method generates a synthetic image of modality $\mathcal{B}$ (right) that is pixel-wise aligned with the map image, but contains the same appearance and observed scenes as the live data image. Top: localising radar data against satellite imagery. Middle: localising lidar data against satellite imagery. Bottom: localising radar data against prior lidar map.

Building on [39], we propose a method for localising against satellite imagery that is self-supervised. The core idea of both approaches is to generate a synthetic image with the appearance and observed scenes of a live range sensor image, but pixel-wise aligned with the satellite image. We assume a coarse initial pose estimate is available from place recognition, such that there is reasonable overlap between the live ground sensor field-of-view and a queried satellite image.

Vitally, here we make no use of metrically accurate ground truth for training. Note also that although designed for localising against satellite imagery, our method can naturally handle other forms of cross-modality registration, such as localising a radar against a prior lidar map. Figure 1 shows synthetic images generated by our method used for pose estimation.

To the best of our knowledge, this paper presents the first method to learn the cross-modality, metric localisation of

a range sensor in a self-supervised fashion. Our method is validated experimentally on multiple datasets and achieves performances on-par with a state-of-the-art supervised approach.

II. Related Work

A. Localisation Using Overhead Images

Localisation using aerial or overhead images has been an interest for the community for over a decade. The methods in [24, 26, 34] localise a ground camera using aerial images, by detecting Canny edges from aerial imagery, and matching against lines detected by a ground camera. Several other vision-based approaches project the ground camera images to a top-down perspective via a homography, and compare against the aerial imagery by detecting lane markings [35], SURF features [32], or dense matching [37]. Recent work [13] localises a ground robot in a crop field by matching camera features against landmarks from an aerial map, and incorporated semantics of crops to reduce ambiguity.

Metric localisation of range sensors or point-clouds against overhead imagery requires further pre-processing due to the modality difference. Kaminsky et al. [19] projected point-clouds into images and matched against binary edge images from overhead imagery. The method in [19] also constructs a ray image by ray-tracing each point, and introduces a free-space cost to aid the image registration. The work by Veronese et al. [15] accumulates several lidar scans to produce dense lidar intensity images, which are then matched against satellite images utilising Normalised Mutual Information. Similar as [19], several other methods also pre-process the aerial image before matching against ground laser observations, for example using edge detection [21] or semantic segmentation [16]. Our method directly learns the metric localisation of a range sensor end-to-end, without the need for careful pre-processing.

B. Cross-Modality Localisation

Other forms of cross-modality localisation are also heavily studied by the community. A number of works are proposed to localise a forward facing camera against a prior 3D point-cloud map [43, 11, 47]. Carle and Barfoot [10] localised a ground laser scanner against an orbital elevation map. The works in [40, 8, 41, 31] localise an indoor lidar or stereo camera against architectural floor plans. Brubaker et al. [9] and Floros et al. [17] concurrently proposed matching visual odometry paths to road layouts from OpenStreetMap for localisation.

C. Learning-based State Estimation for Range Sensors

A number of recent works were proposed for learning odometry or localisation of lidars. Barsan et al. [7] represented lidar data as intensity images, and learned a deep embedding for metric localisation by comparing embeddings of live and map lidar intensity images. Methods such as [14, 27] learn deep lidar odometry by projecting lidar point-clouds into other representations before passing through the network. Lu et al. [30] used point-clouds as input to learn descriptors, and utilised 3D Convolutional Neural Networks (CNNs) for solving $SE(2)$ metric localisation by searching in a 3D cost volume. In their later work, Lu et al. proposed a method to learn $SE(3)$ lidar point-cloud registration end-to-end [29].

As an emerging sensor for outdoor state estimation, learning-based methods were proposed for scanning Frequency-Modulated Continuous-Wave (FMCW) radars. Aldera et al. [3] utilised an encoder-decoder on polar image representation of radar scans to learn key-points for fast classical radar odometry [12]. Barnes et al. learned image-based radar odometry [5] by masking out regions distracting for pose estimation, and point-based radar odometry [4] by detecting key-points from radar images. Saftescu et al. [36] encoded images of radar polar scans through a rotation-invariant architecture to perform topological localisation (place recognition). These methods, however, are designed to compare data of the same sensor type, and does not address modality difference. Our approach is similar to [7, 14, 3, 5, 42, 36, 4] in that we also represent lidar and radar data as 2D images prior to passing through the network.

D. Unsupervised Image Generation

We seek to generate a synthetic image prior to pose computation, where there is no pixel-wise aligned target image for supervision. CycleGAN [48] achieves unsupervised image-to-image transfer between two domains $\mathcal{X}$ and $\mathcal{Y}$, by learning two pairs of generators and discriminators, and enforcing cycle-consistency when an image is mapped from $\mathcal{X}$ to $\mathcal{Y}$ and back from $\mathcal{Y}$ to $\mathcal{X}$, and vice versa. A number of other methods [28, 22] also utilise cycle-consistency, but make different assumptions on how the latent spaces of the two domains are treated. These methods are concerned with generating photo-realistic images. For the problem of metric localisation, however, we need to explicitly encourage the synthetic image to contain information appropriate for pose estimation.

Several prior works are also geometry-aware. The methods in [38, 45, 46] use separate encoders and/or decoders to disentangle geometry and appearance. The results are networks that can interpolate the geometry and appearance of the output images separately. Similarly, our method separately encodes information about appearance and the relative pose offset, resulting in an architecture where the two are disentangled.

III. Overview and Motivation

We seek to solve for the $SE(2)$ pose between a map image of modality $\mathcal{A}$ and a live data image of modality $\mathcal{B}$. Our main focus is when modality $\mathcal{A}$ is satellite imagery, while modality $\mathcal{B}$ are range sensor data represented as an image.

Previously, RSL-Net [39] was proposed to solve for the metric localisation between matched pairs of radar and satellite images. In particular, a synthetic image is generated such that it preserves the appearance and observed scenes of the live radar image, and is pixel-wise aligned with the paired satellite image. The synthetic image and the live radar image are then projected onto deep embeddings, where their pose offset is found by maximising a correlation surface. We follow the same general approach, but, unlike RSL-Net, our method learns in a self-supervised fashion.

A. Hand-crafting Features vs. Learning

A number of works listed in Sections II-B and II-A can achieve decent accuracy on localising a ground range sensor against aerial imagery. However, they typically rely on pre-processing the aerial images using hand-crafted features or transforms designed for a specific set-up and may not generalise to other sensors or different, more complex environments. For example, [21] focuses on detecting edges from a campus dominated by buildings. While [15] directly matches accumulated lidar intensity images against aerial imagery without pre-processing, the same method is unlikely to work for radars.

Our data-driven approach instead learns to directly infer the geometric relationship across modalities, remaining free of hand-crafted features. We show in Section V when localising against satellite imagery, our method works for various types of scenes including urban, residential, campus, and highway.

B. Generating Images vs. Direct Regression

A naive approach would be to take a satellite image and a live data image as inputs, and directly regress the pose. As originally shown in [39], this led to poor results even for the supervised case. Our hypothesis is that when the two images are starkly different in appearance and observed scenes, the problem becomes too complex for direct regression to succeed.

Generating synthetic images first prior to pose estimation brings two advantages over directly regressing the pose. First, generating synthetic images is a simpler and less ill-posed problem than directly regressing the pose, particularly because we can utilise the live data image to condition the generation. Moreover, to generate images, the network loss is distributed over an entire image of $H \times W$ pixels, where H and W are height and width, instead of on just three parameters (x, y, and θ). This introduces greater constraint during optimisation.

C. Conditional Image Generation

Our method of conditional image generation takes in both a map (e.g., satellite) image and a live data image as inputs. An alternative approach is to learn a domain adaptation from the map modality $\mathcal{A}$ to the live data modality $\mathcal{B}$, without conditioning on the live data image (e.g., standard image-to-image transfer such as CycleGAN [48]).

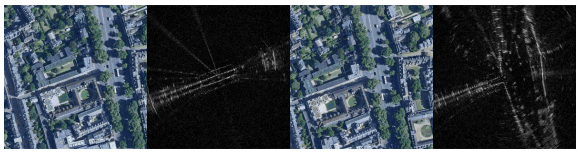


Fig. 2: Two radar images captured 15 seconds apart from each other (2 & 4), pixel-wise aligned with satellite images (1 & 3). Though the overlapping scenes in the satellite images are identical, the radar scans appear significantly different, as they capture different regions in their field-of-view.

In practise, the map (e.g., satellite) image is a denser representation of the environment than a frame of data captured by a range sensor. Only a fraction of the scenes captured in a satellite map are present in a ground sensor field-of-view, resulting in the scan to appear drastically different depending on the sensor pose. Shown in Figure 2, the overlapping regions of the two satellite images are identical, while the two radar images observe different regions of the scene.

By using a naive image-to-image transfer approach, there is no guarantee the generated image will contain regions of the scene that are useful for pose comparison against the live data image. Figure 11 shows examples of images generated using CycleGAN [48], where the synthetic image highlights different scenes than what are observed by the live data image. The issue with observability or occlusion can potentially be handled by ray-tracing such as in [19]. However, not only is this computationally expensive, it does not apply to FMCW radars which have multiple range returns per azimuth (in which we are particularly interested). This problem is inherently addressed by our approach: by conditioning the image generation with the live data image, we can encourage the synthetic image to capture regions of the scene also observed by the live data image, as shown in Sections IV and V.

IV. Self-Supervised Cross-Modality Localisation

A. Rotation Inference

Given a paired map (e.g., satellite) image $A \in \mathcal{A}$ and live data image $B \in \mathcal{B}$ with an unknown $SE(2)$ offset, we seek to generate a synthetic image that contains the same appearance and observed scenes as B, but is pixel-wise aligned with A.

Let the $SE(2)$ pose difference between A and B be parametrised as $\begin{bmatrix} x & y & \theta \end{bmatrix}^T$, such that by rotating B by θ, followed by a translation of $\begin{bmatrix} x & y \end{bmatrix}^T$, one can pixel-wise align B onto A. The image generation can be formulated as:

$$f(A, B) \rightarrow \tilde{B}_{\theta,\alpha} \tag{1}$$

where $\alpha = \begin{bmatrix} x & y \end{bmatrix}^T$. $\tilde{B}_{\theta,\alpha}$ is a generated image of modality $\mathcal{B}$ that synthesises the input live sensor image B applied with a rotation of θ, followed by a translation of $\alpha = \begin{bmatrix} x & y \end{bmatrix}^T$. Thus, $\tilde{B}_{\theta,\alpha}$ is pixel-wise aligned with the input map image A, but contains the same observed scenes as B.

However, as originally noted in [39], the mapping in (1) is difficult to learn as the inputs A and B are offset by both a translation and a rotation. CNNs are inherently *equivariant* to translation, but not to rotation [23]. As a result, the CNNs in the generator cannot automatically utilise their mutual information and thereby capture their geometric relationship.

The method in [39] proposes to infer the rotation prior to image generation. Namely, reducing (1) to two steps:

$$f_R(A, B) \rightarrow B_\theta \tag{2}$$

$$f_G(A, B_\theta) \rightarrow \tilde{B}_{\theta,\alpha} \tag{3}$$

Here f_R is a function that infers the rotation offset θ between A and B, and outputs B_θ, which is input image B rotated by θ. Now, B_θ is rotation-aligned with the map frame, and therefore offset with A only by a translation, which CNNs can naturally handle. f_G is an image generation function that produces the synthetic image $\tilde{B}_{\theta,\alpha}$. The experiments in [39] show that learning (2) and (3) sequentially resulted in

better performance than learning (1) directly, as the former is congruous with the equivariance properties of CNNs.

In [39], the rotation inference function f_R is parametrised by a deep network as shown in Figure 3, where satellite imagery and radar images are used as an example. Given a coarse initial heading estimate, the live data image B is rotated a number of times with small increments to form a stack of rotated images $\{B\} = \{B_{\theta_0}, B_{\theta_1}, \ldots, B_{\theta_n}\}$, where the number of rotations n and the increment are design parameters. Each rotated image is further concatenated with the map image to form a stacked tensor input of n pairs of map and live data images. The output of the network is a softmaxed image from $\{B\}$ that corresponds to B rotated to be rotation-aligned with A, namely B_θ. The core idea is that the network f_R will assign a large softmax weight to the image from $\{B\}$ whose heading most closely aligns with the map image A, and small weights to all other images in $\{B\}$.

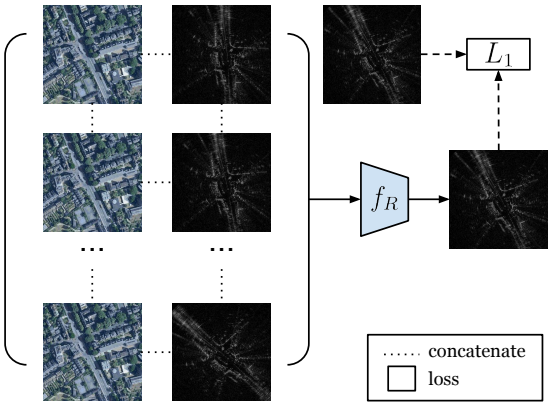


Fig. 3: Prior work in [39] proposes a network to infer the rotation offset. The rotation offset is found by softmaxing a stack of rotated radar images to produce a radar image with the same heading as the satellite image.

If metrically accurate heading ground truth θ is available, then one can rotate B to form an image target to B_θ used for supervising the rotation inference, as in Figure 3. In this work we assume this is never the case, thus the network for f_R must learn to infer the rotation offset self-supervised.

For this reason, while following the same architecture as [39], our method for inferring rotation uses a different training strategy that enables self-supervised learning. In order for the network f_R to produce the correct output, it must be able to infer the rotation from the solution space $\{B\}$, despite there being a modality difference between map image A and live data image B. We make the observation that if the network can infer the rotation offset from a stack of rotated live data images $\{B\}$, then, given a live data image B_{θ_i}, f_R should also be able to output A_{θ_i} from a stack of rotated map images $\{A\}$, where A_{θ_i} is rotation-aligned with B_{θ_i}. Specifically, if we have $B_{\theta_i} = B_\theta$, then the softmaxed map image from $\{A\}$ should be A, as A and B_θ are rotation-aligned.

As such, to learn rotation inference self-supervised, we need to pass through the network f_R twice. The first pass is identical as in the supervised approach in Figure 3, where we denote the output softmaxed image as B_{θ_i}. B_{θ_i} is then used as input to the second pass through network f_R, together with a stack of map images $\{A\} = \{A, A_{\phi_0}, A_{\phi_1}, \ldots, A_{\phi_m}\}$. The rotation angles $\begin{bmatrix}\phi_0 & \phi_1 & \ldots & \phi_m\end{bmatrix}$ can be chosen randomly, and the order of $\{A\}$ is shuffled such that the original non-rotated map image A can be at any index within $\{A\}$. Each image is concatenated with B_{θ_i} to form the input stack for passing through f_R the second time. The network is supervised with an L_1 loss that enforces the output of the second pass to be the non-rotated map image A, which in turn enforces the output of the first pass B_{θ_i} to be B_θ, as B_θ is rotation-aligned with A. Our approach is shown in Figure 4. We use an increment of 2° when forming the rotation stack $\{B\}$.

The estimate for the rotation offset, $\hat{\theta}$, can then be found from the arg-softmax for the rotation stack $\{B\}$.

B. Image Generation

Given A and B_θ we seek to generate a synthetic image $\tilde{B}_{\theta,\alpha}$ as in (3), where $\tilde{B}_{\theta,\alpha}$ is pixel-wise aligned with A. [39] learns the image generation function by a supervised approach, concatenating A and B_θ, and applying an encoder-decoder architecture, as shown in Figure 5. This is possible since a target for the synthetic image $\tilde{B}_{\theta,\alpha}$ can be obtained by applying the ground truth transform.

To generate synthetic images self-supervised, we propose an architecture we call Pose-Aware Separable Encoder Decoder (PASED), shown in Figure 6. PASED is trained in two steps: the first is a pre-training, intra-modality process that can be supervised (top half of Figure 6), while the second handles cross-modality comparison (bottom half of Figure 6).

Taking two random images B^1 and B^2 in the live data modality $\mathcal{B}$ from the training set, where B^1 and B^2 can be at arbitrary heading, we apply a known translation offset $\gamma \in \mathbb{R}^2$ to B^2. This forms an image B^2_γ that is a shifted version of B^2. We pass B^1 through an appearance encoder E_a that encodes its appearance and observed scenes. B^2_γ and B^2 are passed as inputs to a pose encoder E_p that encodes the translation offset between the input images. The latent spaces from E_a and E_p are combined before passing through a decoder D, which outputs a synthetic image $\tilde{B}^1_\gamma$ that is B^1 shifted by a translation γ. In other words, PASED discovers the translation offset between the two images passed as input to E_p, and applies the latent translation encoding to the input image of E_a. The pre-training can be supervised as γ is known, thus we can shift B^1 by γ to produce the target B^1_γ. The fact that we use different images B^1 and B^2 for inputs to E_a and E_p ensures appearance and pose are disentangled from each other. As shown later, this allows modules of PASED to be separated and re-combined with newly learned modules.

In the second step, we fix the weights of E_a, E_p, and D which are optimised from the pre-training step. This narrows down the self-supervision problem to learning a cross-modality pose encoder E^*_p that discovers the translation offset between an image of modality $\mathcal{A}$ and another of $\mathcal{B}$. Taking A and B_θ as inputs, E^*_p should encode the unknown translation offset α between them. Concurrently, B_θ is encoded by E_a, where the latent space is combined with the latent space produced

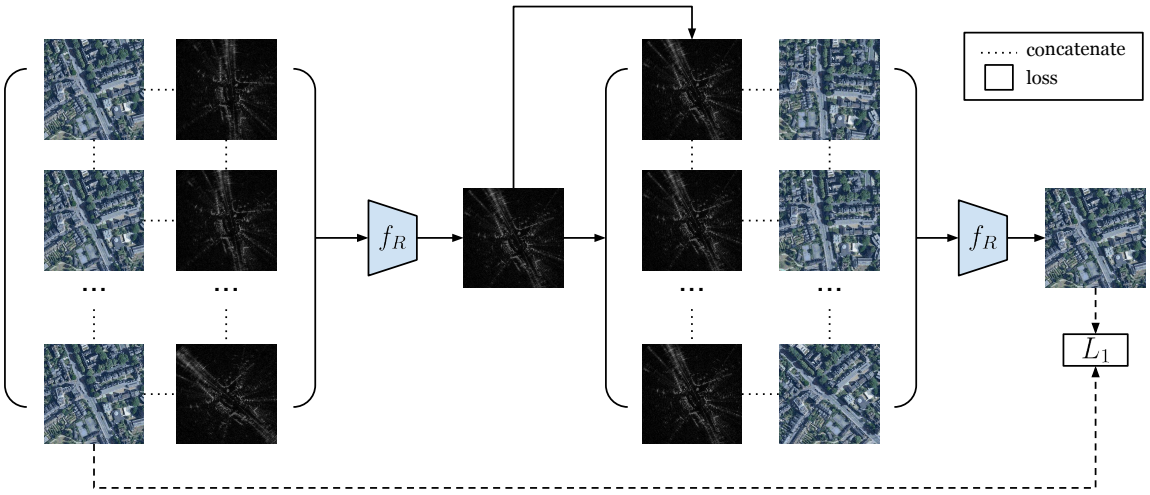


Fig. 4: Given A and a rotation stack $\{B\}$ the network f_R finds B_{θ_i} by taking softmax. Then, given B_{θ_i} and a rotation stack $\{A\}$, the network outputs a softmaxed map image from $\{A\}$. A loss is applied to enforce the output of the second pass to be A, which in turn enforces the output of the first pass to be B_θ. Here both symbols for f_R in the figure refer to the same network, but at different forward passes.

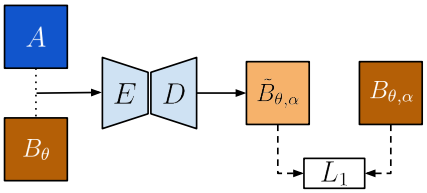


Fig. 5: Architecture for image generation in prior supervised approach [39].

by $E_p^*(A, B_\theta)$, before decoded by D. This encoder-decoder combination will generate a synthetic image $\tilde{B}_{\theta,\alpha}$, which we do not have a target for.

We can apply a known shift to the centre position of A to query another map image A', where A' is offset with B_θ by an unknown translation β. Using the same encoder-decoder combination as before, we can take A' and B_θ to generate a synthetic image $\tilde{B}_{\theta,\beta}$. Furthermore, given B_θ and the networks learned from pre-training, we can easily generate $\tilde{B}_\theta$ by encoding a zero shift. If we pass $\tilde{B}_\theta$ and $\tilde{B}_{\theta,\alpha}$ to the pre-trained pose encoder E_p, then the latent space will encode a shift of $-\alpha$. Combing this latent space with $E_a(\tilde{B}_{\theta,\beta})$, we can decode a synthetic image $\tilde{B}_{\theta,\beta-\alpha}$. Here $\beta - \alpha$ is a known value as it is the translation offset applied to A to obtain A'.

We can shift B_θ by $\beta - \alpha$ to get $B_{\theta,\beta-\alpha}$. Using $B_{\theta,\beta-\alpha}$ and B_θ, we can generate $\tilde{B}_{\theta,\beta-\alpha}$ with E_a, E_p, and D, shown on the bottom right of Figure 6. A loss can then be established between the two synthetic images $\tilde{B}_{\theta,\beta-\alpha}$, where the latter one is a target image created by modules with weights fixed. By back-propagation the loss optimises the network E_p^*. Alternatively we can use $B_{\theta,\beta-\alpha}$ as the target, but using $\tilde{B}_{\theta,\beta-\alpha}$ led to faster convergence.

For the loss to be minimised, two conditions must hold true. First, $\tilde{B}_{\theta,\beta}$ must have correctly encoded the appearance and observed scenes in B_θ. Second, $\tilde{B}_{\theta,\alpha}$ and $\tilde{B}_{\theta,\beta}$ must have the correct translations α and β, respectively. By satisfying these two constraints we can ensure E_p^* is able to discover the translation offset across modalities, and is compatible with pre-trained networks E_a and D for image generation.

C. Pose Estimation

Taking $\tilde{B}_{\theta,\alpha}$ and B_θ, we embed them to a joint space, where their translation offset is found by maximising correlation on the learned embeddings. This can be performed efficiently in the Fourier domain, as is done in prior works that use a similar approach [7, 5, 39]. In this step, we can infer $\hat{\alpha} = \begin{bmatrix}\hat{x} & \hat{y}\end{bmatrix}^T$, which is our posterior estimate to the translation.

The embeddings are thus learned to further ensure the synthetic image and the live image can be correctly correlated. Without ground truth α, we can self-supervise using a similar approach as in learning PASED, by applying a known shift. The architecture for learning the embeddings is shown in Figure 7, where we denote the embedding network for real and synthetic images to be H_B and $H_{\tilde{B}}$, respectively. Given learned deep embeddings $\tilde{B}^\dagger_{\theta,\beta}$ and $B^\dagger_\theta$, the translation offset by correlation maximisation is found to be $\hat{\beta}$. If we replace $\tilde{B}^\dagger_{\theta,\beta}$ with $\tilde{B}^\dagger_{\theta,\alpha}$ and reverse the order, the offset found will be $-\hat{\alpha}$. The sum of the two offsets $\beta - \alpha$ is known, and can be used to establish a loss term. Similar as in Section IV-B, $\tilde{B}_{\theta,\beta}$ is obtained by shifting the map image A to get A'.

V. Experimental Validation

The overall pipeline for data flow at inference time is shown in Figure 8. The inference runs at about 10 Hz on a single 1080 Ti GPU. We evaluate on a large number of public, real-world datasets collected with vehicles equipped with on-board range sensors. The datasets we use come with metric ground truths that are decently accurate, though we noticed the GPS/INS solutions in certain places can drift up to a few metres.

We add large artificial pose offsets to the ground truth when querying for a satellite image, thereby simulating a realistic robot navigation scenario where the initial pose estimate can solve place recognition, but is too coarse for the robot's metric pose. Using a map (e.g., satellite) image queried at this coarse

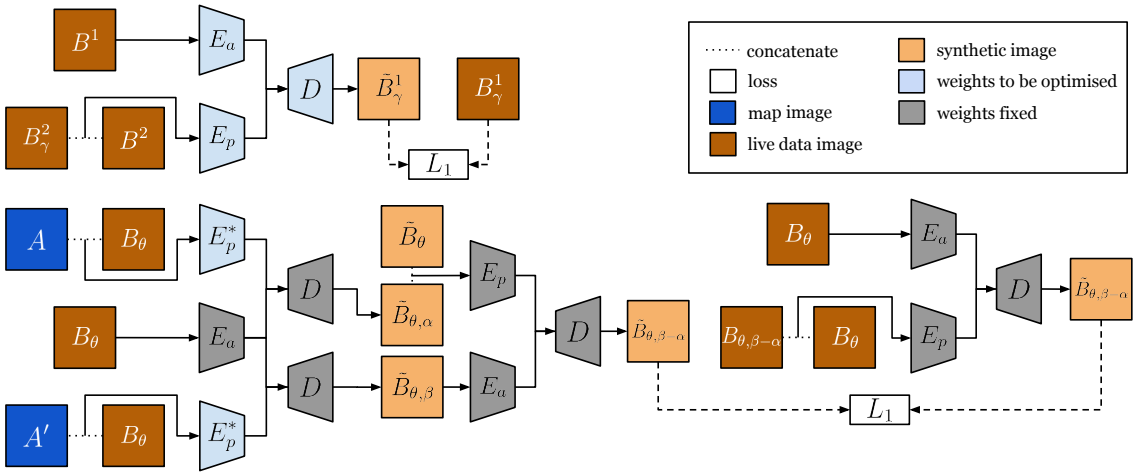


Fig. 6: Top: during pre-training, we can learn an appearance encoder E_a, and a pose encoder E_p that discovers the translation offset between an image of $\mathcal{B}$ and a shifted version of itself. Bottom: Taking E_a, E_p, and D and fixing their weights, we seek to learn E_p^* which discovers the translation offset between two images from different modalities. E_a, E_p, and D can provide the necessary geometric and appearance relationships used for learning E_p^* self-supervised.

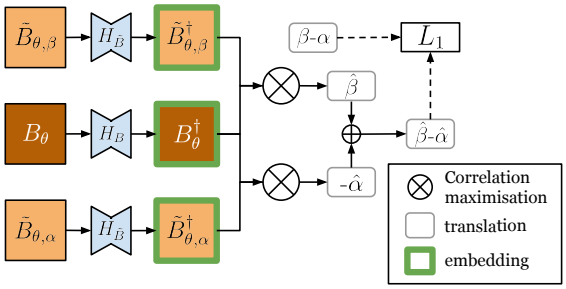


Fig. 7: The networks H_B and $H_{\tilde{B}}$ are learned to project real live images and synthetic images to a joint embedding, where their translation offset can be found by maximising correlation.

initial pose estimate, our method solves metric localisation by comparing against the live sensor data. The true pose offsets are hidden during training as our method is self-supervised, and are only revealed at test time for evaluation purposes.

The artificial offset is chosen such that the initial estimate has an unknown heading error in the range $[-\frac{\pi}{8}, \frac{\pi}{8}]$, therefore given the initial estimate θ_0, the rotation inference must choose a solution space of at least $[\theta_0 - \frac{\pi}{8}, \theta_0 + \frac{\pi}{8}]$ to guarantee the correct solution can be found. We use a pixel-wise translation error in the range $[-25, 25]$ pixels. Depending on the resolution for a specific experiment, this corresponds to an error of at least $[-10\text{m}, 10\text{m}]$ and up to more than $[-20\text{m}, 20\text{m}]$.

A. Radar Localisation Against Satellite Imagery

We evaluate on two datasets with FMCW radar and GPS: the Oxford Radar RobotCar Dataset [6] and the MulRan Dataset [20]. The satellite images for RobotCar are queried using Google Maps Platform [2]. For MulRan they are queried using Bing Maps Platform [1], as high-definition Google satellite imagery is unavailable at the place of interest.

We benchmark against the prior supervised method RSL-Net [39] in our experiments, which is evaluated only on the RobotCar Dataset. Both datasets contain repeated traversals of the same routes. We separately train, validate, and test for every dataset, splitting the data as in Figure 9. For the RobotCar Dataset, we split the trajectories the same way as in [39] for a fair comparison. For the RobotCar Dataset, the training set consists of training data from sequences no. 2, no. 5, and no. 6, while we test on the test data from sequence no. 2. For the MulRan Dataset, we used sequences `KAIST 01` and `Sejong 01`. The RobotCar test set features an urban environment, while `KAIST 01` is in a campus and `Sejong 01` is primarily a highway.

	Mean Error (metric)			(pixel)	
	x (m)	y (m)	θ (°)	x	y
RobotCar (ours)	3.44	5.40	3.03	3.97	6.23
MulRan (ours)	6.02	7.02	2.92	7.64	8.91
RobotCar (RSL-Net [39], supervised)	2.74	4.26	3.12	3.16	4.92
MulRan (RSL-Net)	5.85	7.11	1.88	7.42	9.03

TABLE I: Mean error for radar localisation against satellite imagery.

We test on every fifth frame, resulting in 201 frames from the RobotCar Dataset and 358 from the MulRan Dataset, spanning a total distance of near 4 km. The resolution used is 0.8665 m/pixel for RobotCar and 0.7876 m/pixel for MulRan. The mean errors are reported in Table I.

B. Lidar Localisation Against Satellite Imagery

For this experiment, we evaluate on the RobotCar Dataset [6] which also has two Velodyne HDL-32E lidars mounted in a tilted configuration, and KITTI (raw dataset) [18] which has a Velodyne HDL-64E lidar and GPS data.

For the RobotCar Dataset, the trajectories are split into training, validation, and test sets approximately the same way as in Section V-A. For the KITTI Dataset, the training set includes sequences `20110929_drive0071`, `20110930_drive0028`, and `20111003_drive0027`. Sequence `20110926_drive0117` is used for validation. Finally, data in `20111003_drive0034` are split into training and test, as shown in Figure 9. To turn 3D lidar point-clouds to lidar images, the point-clouds are projected to the $x - y$

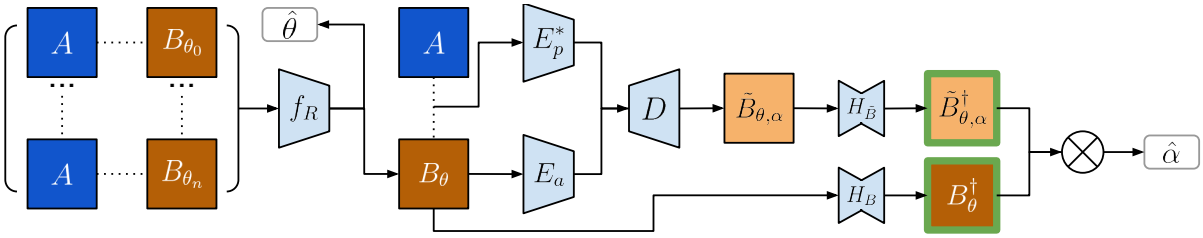


Fig. 8: Overall data flow of our method at inference: given map image A and live data image B, based on the initial heading estimate, we form a stack of rotated images $\{B_{\theta_0}, \ldots, B_{\theta_n}\}$, from which f_R discovers B_θ that is B rotated to be rotation-aligned with A. This process also infers the heading estimate $\hat{\theta}$. A and B_θ are used to generate a synthetic image $\tilde{B}_{\theta,\alpha}$ that has the same appearance and observed scene as B_θ and is pose-aligned with A. $\tilde{B}_{\theta,\alpha}$ and B_θ are projected to deep embeddings $\tilde{B}^\dagger_{\theta,\alpha}$ and $B^\dagger_\theta$, where the estimate for the translation offset $\hat{\alpha}$ is found by correlation maximisation.

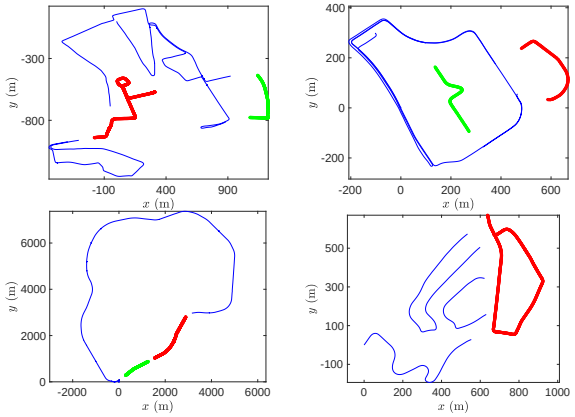


Fig. 9: Training (blue), validation (green), and test (red) trajectories for RobotCar (top left), KAIST (top right), Sejong (bottom left) and 20111003_drive0034 (bottom right). Certain data are removed to avoid overlap between the splits.

plane. We discard points with z values smaller than zero to remove ground points when creating the lidar images.

Since lidars have a shorter range than radars, we use satellite images of a greater zoom level, with resolution 0.4332 m/pixel for RobotCar and 0.4592 m/pixel for KITTI. The test set consists of 200 frames for RobotCar and 253 for KITTI, spanning a total distance of near 3 km. The test set for KITTI features a residential area. The results are reported in Table II.

	Mean Error (metric)			**(pixel)**	
	x (m)	y (m)	θ (°)	x	y
RobotCar (ours)	1.54	1.85	2.34	3.55	4.27
KITTI (ours)	3.05	3.13	1.67	6.64	6.82
RobotCar (RSL-Net)	2.31	2.55	2.08	5.33	5.89
KITTI (RSL-Net)	2.45	2.79	1.59	5.34	6.08

TABLE II: Mean error for lidar localisation against satellite imagery.

C. Radar Localisation Against Prior Lidar Map

	Mean Error (metric)			**(pixel)**	
	x (m)	y (m)	θ (°)	x	y
RobotCar (ours)	2.21	2.57	2.65	2.55	2.97
MulRan (ours)	3.57	3.26	2.15	4.53	4.13
RobotCar (RSL-Net)	2.66	3.41	2.45	3.07	3.93
MulRan (RSL-Net)	3.37	2.61	1.40	4.28	3.32
RobotCar (CycleGAN)	6.41	9.05	2.65	7.40	10.44
MulRan (CycleGAN)	4.84	4.39	2.15	6.14	5.58

TABLE III: Mean error for radar localisation against prior lidar map.

Though our method is designed for localising against satellite imagery, we show it can also handle more standard forms of cross-modality localisation. Here we build a lidar map using a prior traversal, and localise using radar from a later traversal.

We demonstrate on the RobotCar and MulRan datasets, where we use the same resolution as in Section V-A. For RobotCar, we use ground truth to build a lidar map from sequence no. 2. Radar data in the training sections from no. 5 and no. 6 as in Figure 9 form the training set, while the test section from sequence no. 5 forms the test set. For MulRan, lidar maps are built from KAIST 01 and Sejong 01, and we localised using radar data from KAIST 02 and Sejong 02, which are split into training, validation, and test sets. This resulted in a test set consisting of 201 frames from RobotCar and 272 frames from MulRan, spanning a total distance of near 4 km. The localisation results are shown in Table III.

This experiment is more suitable for naive image generation methods such as CycleGAN [48] than previous experiments, as the field-of-view is considerably more compatible when both modalities are from range sensors. In Table III, we show results where we replaced the image generation stage of our method by CycleGAN, and kept other modules. The localisation results are however much worse when modality $\mathcal{A}$ is satellite imagery, as shown qualitatively in Figure 11.

D. Online Pose-Tracking System

In prior experiments we assumed place recognition is always available, providing a coarse initial estimate for every frame. Here we present a stand-alone pose-tracking system by continuously localising against satellite imagery. Given a coarse initial estimate (e.g., from GPS) for the first frame, the vehicle localises and computes its pose within the satellite map. The initial estimate for every frame onward is then set to be the computed pose of the previous frame. We only need place recognition once at the very beginning; the vehicle then tracks its pose onward without relying on any other measurements.

1) Introspection: As localising using satellite imagery is challenging, the result will not always be accurate. Our method, however, naturally allows for introspection. A synthetic image $\tilde{B}_{\theta,\alpha}$ was generated from A and B_θ. We can apply a known small translation offset δ to A to form A_δ. Taking A_δ and B_θ we can generate $\tilde{B}_{\theta,\alpha+\delta}$. Finally, we can compute a translation offset $\hat{\delta}$ by passing $\tilde{B}_{\theta,\alpha+\delta}$ and $\tilde{B}_{\theta,\alpha}$ through the learned embeddings and maximising correlation.

Let $d_{\text{intro}} = \left\| \delta - \hat{\delta} \right\|$. A large value of d_{intro} indicates the generated images are erroneous. This allows us to examine

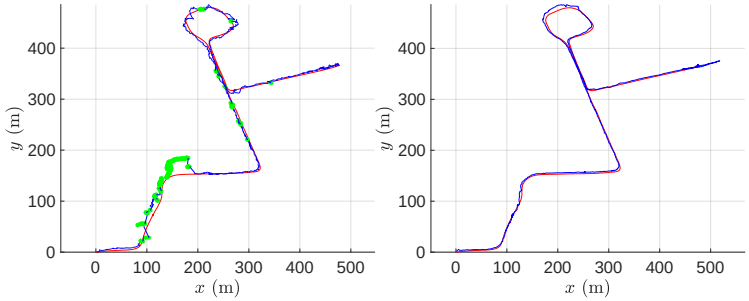


Fig. 10: Estimated pose (blue) vs. ground truth pose (red) for localising a radar (left) and a lidar (right) against satellite imagery. Our system continuously tracks the vehicle's pose over 1 km, where we occasionally fall back to odometry for the radar experiment (green). Our system is stand-alone and requires GPS only for the first frame.

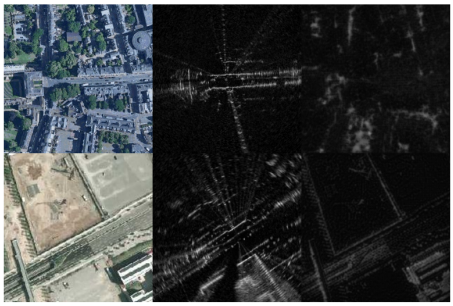
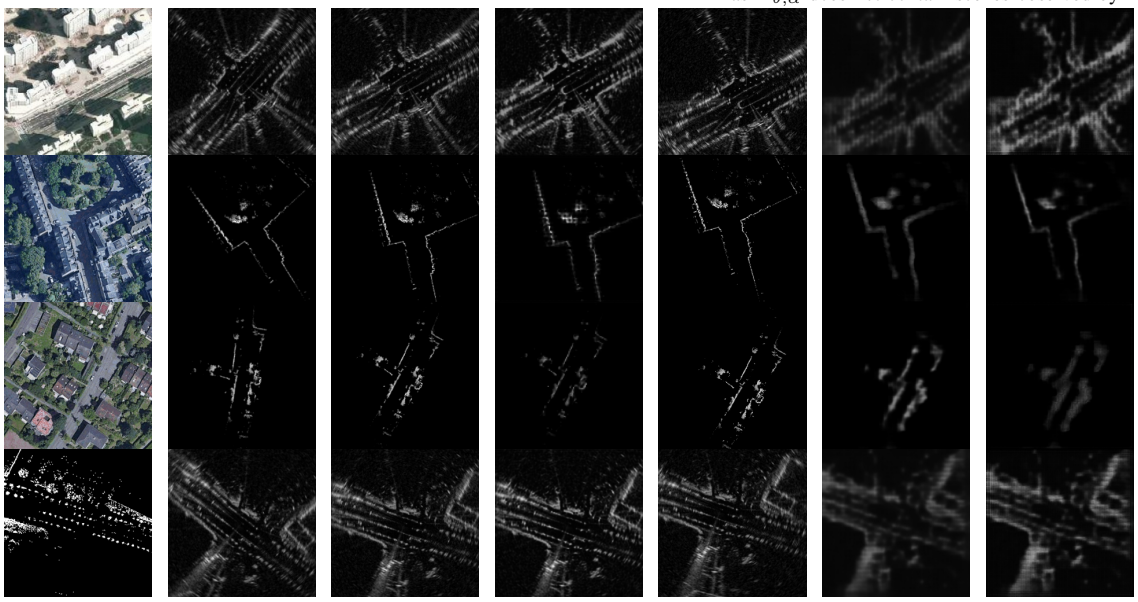


Fig. 11: Results of CycleGAN: satellite image A (left), ground truth radar image $B_{\theta,\alpha}$ (middle), synthetic radar image $\tilde{B}_{\theta,\alpha}$ (right). This led to large localisation error as $\tilde{B}_{\theta,\alpha}$ does not contain scenes observed by B.



(a) (b) (c) (d) (e) (f) (g)

Fig. 12: Images at various stages of our method: map image A (a), live data image B (b), output of rotation inference B_θ (c), embedding $B_\theta^\dagger$ (d), pixel-wise aligned ground truth $B_{\theta,\alpha}$ (e), synthetic image $\tilde{B}_{\theta,\alpha}$ (f), embedding$\tilde{B}_{\theta,\alpha}^\dagger$ (g). From top to bottom: radar localisation against satellite imagery evaluated on MulRan, lidar localisation against satellite imagery evaluated on RobotCar and KITTI, radar localisation against lidar map evaluated on MulRan.

the solution quality; our system falls back to using odometry for dead-reckoning when d_{intro} exceeds a threshold. We do not require high-quality odometry, but rather only use a naive approach by directly maximising correlation between two consecutive frames without any learned modules. In our experiments, we set δ to be $\begin{bmatrix}10 & 10\end{bmatrix}^T$, and d_{intro} to be 5.

2) Results: We conduct two experiments on the test set of RobotCar, one where we track a radar using satellite imagery, and one where we track a lidar. For both experiments we run localisation at 4 Hz. The results are shown in Figure 10. If the solution error is too large, then the initial estimate will be too off for a sufficient overlap between the next queried satellite image and live data, resulting in losing track of the vehicle. Though the solution error can be larger than 10 m at times, our system continuously localises the vehicle for over a kilometre without completely losing track. For the lidar experiment, the solutions are sufficiently accurate to not require any odometry. Our experiments are single-frame localisations, and we make no attempt at windowed/batch optimisation or loop closures.

E. Further Qualitative Results

Additional qualitative results are presented in Figure 12 showing various stages of our methods for different modalities.

VI. Conclusion and Future Work

We present self-supervised learning to address cross-modality metric localisation between satellite imagery and on-board range sensors, without using metrically accurate ground truth for training. Our method is validated across a large number of experiments for multiple modes of localisation, with results on-par with prior supervised approach. A coarse initial pose estimate is needed for our method to compute metric localisation. An extension would then be to solve place recognition for a range sensor within a large satellite map.

Acknowledgments

We thank Giseop Kim from IRAP Lab, KAIST for providing GPS data for the MulRan Dataset.

REFERENCES

[1] Bing Maps. https://docs.microsoft.com/en-us/bingmaps/.

[2] Google Maps Platform. https://developers.google.com/maps/documentation/maps-static/intro/.

[3] Roberto Aldera, Daniele De Martini, Matthew Gadd, and Paul Newman. Fast Radar Motion Estimation with a Learnt Focus of Attention Using Weak Supervision. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1190–1196. IEEE, 2019.

[4] Dan Barnes and Ingmar Posner. Under the Radar: Learning to Predict Robust Keypoints for Odometry Estimation and Metric Localisation in Radar. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Paris, 2020. URL https://arxiv.org/abs/2001.10789.

[5] Dan Barnes, Rob Weston, and Ingmar Posner. Masking by Moving: Learning Distraction-Free Radar Odometry from Pose Information. In *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/pdf/1909.03752.

[6] 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 *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Paris, 2020. URL https://arxiv.org/abs/1909.01300.

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

[8] Federico Boniardi, Tim Caselitz, Rainer Kümmerle, and Wolfram Burgard. Robust LiDAR-based Localization in Architectural Floor Plans. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3318–3324. IEEE, 2017.

[9] 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.

[10] 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.

[11] Tim Caselitz, Bastian Steder, Michael Ruhnke, and Wolfram Burgard. Monocular Camera Localization in 3D Lidar Maps. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1926–1931. IEEE, 2016.

[12] Sarah H Cen and Paul Newman. Radar-Only Ego-Motion Estimation in Difficult Settings via Graph Matching. *arXiv preprint arXiv:1904.11476*, 2019.

[13] Nived Chebrolu, Philipp Lottes, Thomas Läbe, and Cyrill Stachniss. Robot Localization Based on Aerial Images for Precision Agriculture Tasks in Crop Fields. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1787–1793. IEEE, 2019.

[14] Younggun Cho, Giseop Kim, and Ayoung Kim. DeepLO: Geometry-Aware Deep LiDAR Odometry. *arXiv preprint arXiv:1902.10562*, 2019.

[15] 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.

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

[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] Ryan S Kaminsky, Noah Snavely, Steven M Seitz, and Richard Szeliski. Alignment of 3D Point Clouds to Overhead Images. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 63–70. IEEE, 2009.

[20] Giseop Kim, Yeong Sang Park, Younghun Cho, Jinyong Jeong, and Ayoung Kim. MulRan: Multimodal Range Dataset for Urban Place Recognition. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. Submitted. Under Review.

[21] 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.

[22] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse Image-to-Image Translation via Disentangled Representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 35–51, 2018.

[23] 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.

[24] 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.

[25] Jesse Levinson and Sebastian Thrun. Robust Vehicle Localization in Urban Environments Using Probabilistic Maps. In *2010 IEEE International Conference on Robotics and Automation*, pages 4372–4378. IEEE, 2010.

[26] Ang Li, Vlad I Morariu, and Larry S Davis. Planar Structure Matching Under Projective Uncertainty for Geolocation. In *European Conference on Computer Vision*, pages 265–280. Springer, 2014.

[27] Qing Li, Shaoyang Chen, Cheng Wang, Xin Li, Chenglu Wen, Ming Cheng, and Jonathan Li. LO-Net: Deep Real-time Lidar Odometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8473–8482, 2019.

[28] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised Image-to-Image Translation Networks. In *Advances in Neural Information Processing Systems*, pages 700–708, 2017.

[29] Weixin Lu, Guowei Wan, Yao Zhou, Xiangyu Fu, Pengfei Yuan, and Shiyu Song. DeepVCP: An End-to-End Deep Neural Network for Point Cloud Registration. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[30] Weixin Lu, Yao Zhou, Guowei Wan, Shenhua Hou, and Shiyu Song. L3-Net: Towards Learning Based LiDAR Localization for Autonomous Driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6389–6398, 2019.

[31] Malcolm Mielle, Martin Magnusson, and Achim J Lilienthal. The Auto-Complete Graph: Merging and Mutual Correction of Sensor and Prior Maps for SLAM. *Robotics*, 8(2):40, 2019.

[32] 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.

[33] Yeong Sang Park, Jinyong Jeong, Youngsik Shin, and Ayoung Kim. Radar Dataset for Robust Localization and Mapping in Urban Environment . In *ICRA Workshop on Dataset Generation and Benchmarking of SLAM Algorithms for Robotics and VR/AR*, Montreal, May. 2019.

[34] Martin P Parsley and Simon J Julier. Towards the Exploitation of Prior Information in SLAM. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2991–2996. IEEE, 2010.

[35] 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.

[36] Stefan Saftescu, Matthew Gadd, Daniele De Martini, Dan Barnes, and Paul Newman. Kidnapped Radar: Topological Radar Localisation using Rotationally-Invariant Metric Learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Paris, 2020. URL https://arxiv.org/abs/2001.09438.

[37] 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.

[38] Zhixin Shu, Mihir Sahasrabudhe, Riza Alp Guler, Dimitris Samaras, Nikos Paragios, and Iasonas Kokkinos. Deforming Autoencoders: Unsupervised Disentangling of Shape and Appearance. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 650–665, 2018.

[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] Xipeng Wang, Ryan J Marcotte, and Edwin Olson. GLFP: Global Localization from a Floor Plan.

[41] Xipeng Wang, Steve Vozar, and Edwin Olson. FLAG: Feature-based Localization between Air and Ground. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3178–3184. IEEE, 2017.

[42] 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.

[43] Ryan W Wolcott and Ryan M Eustice. Visual Localization within Lidar Maps for Automated Urban Driving. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 176–183. IEEE, 2014.

[44] 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.

[45] Wayne Wu, Kaidi Cao, Cheng Li, Chen Qian, and Chen Change Loy. Transgaga: Geometry-Aware Unsupervised Image-to-Image Translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8012–8021, 2019.

[46] Xianglei Xing, Tian Han, Ruiqi Gao, Song-Chun Zhu, and Ying Nian Wu. Unsupervised Disentangling of Appearance and Geometry by Deformable Generator Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10354–10363, 2019.

[47] Yuquan Xu, Vijay John, Seiichi Mita, Hossein Tehrani, Kazuhisa Ishimaru, and Sakiko Nishino. 3D Point Cloud Map Based Vehicle Localization Using Stereo Camera. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 487–492. IEEE, 2017.

[48] 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.

C

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

This article was published as a conference proceeding at Robotics: Science and System XVII in July 2021.

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].

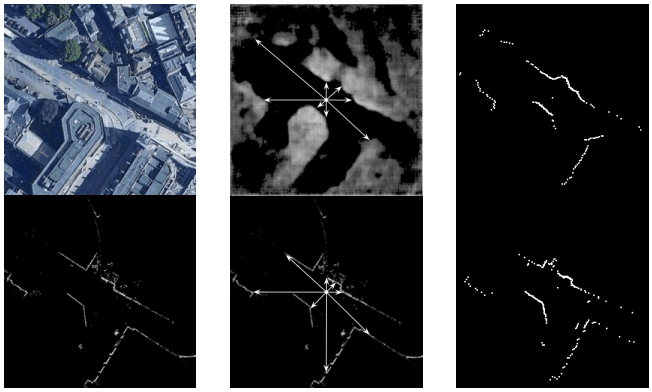


Fig. 1: Top row: given an overhead image (left), we learn an occupancy image (centre) that indicates how likely each pixel belongs to a space that will produce a point return if scanned by a lidar. This results in high probability for buildings and low probabilities for streets and roads. Ray-tracing 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 $\psi\big(f(\chi)\big) = f\big(\phi(\chi)\big), \forall \chi \in \mathcal{X}$.

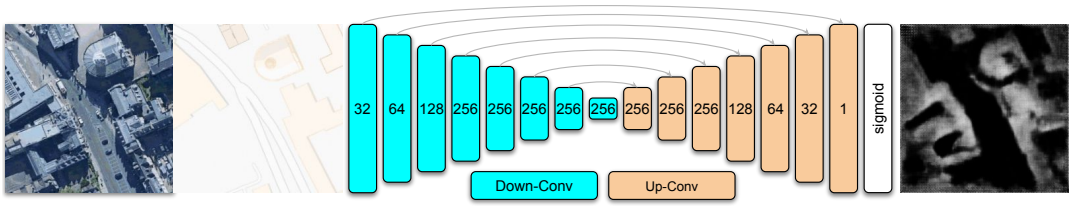


Fig. 3: To generate the occupancy image $\hat{\mathrm{O}}$, we use an U-Net architecture with 8 down-sample convolution blocks (each followed by Leaky ReLU and BatchNorm) and 8 up-sample convolution blocks (each followed by ReLU, BatchNorm, and dropout, except the last block), with skip connections between 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.

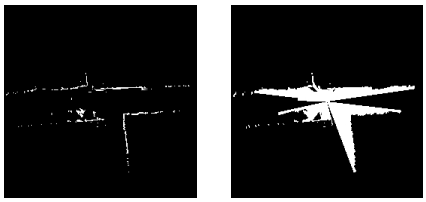


Fig. 4: Given a lidar image I^L we first threshold I^L to produce a binary lidar image $\mathrm{I}^{L\dagger}$ (left) where $\mathrm{I}^{L\dagger}(i,j)$ is 0 or 1 for any pixel $[i,j]$, with 1 indicating return and 0 indicating no return. Ray-tracing $\mathrm{I}^{L\dagger}$, pixels from the centroid 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.

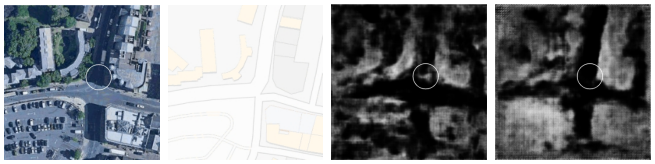


Fig. 5: Left: an unseen satellite image I^S at inference. Centre left: the corresponding I^R queried at the same location as I^S. The circled region corresponds to a strong gradient in I^S caused by shadows, and results in some pixels in this region inferred as occupied where they should be on free-space 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
γ // 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.

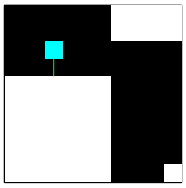


Fig. 6: A 10×10 image patch where the centroid is on occupied space, which are labelled as white pixels. For a particular pixel in this patch (cyan), we can compute its distance (green) to the nearest occupied pixel. The pixel within the patch that is the furthest away from its nearest occupied pixel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

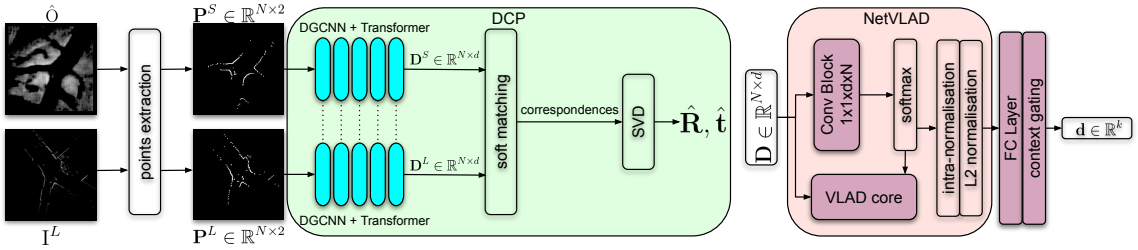


Fig. 7: Data flow at forward pass. Left: for pose estimation, given a spatially proximal pair of $\hat{\mathrm{O}}$ and I^L, we extract point-clouds $\mathbf{P}^S$ and $\mathbf{P}^L$ respectively, using Algorithm 1. The DGCNN+Transformer module in DCP learns local descriptors for each point-cloud, which 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.

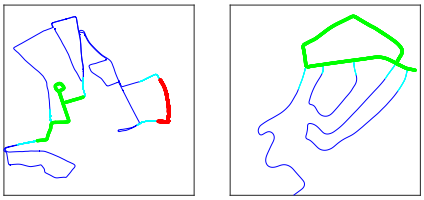


Fig. 8: Trajectories are split into training (blue), validation (red), and test (green) for all sequences in RobotCar (left) and `20111003_drive0034` in KITTI (right). Regions near the intersections between splits are discarded (cyan) to avoid any overlap between training, validation, and test trajectories.

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.

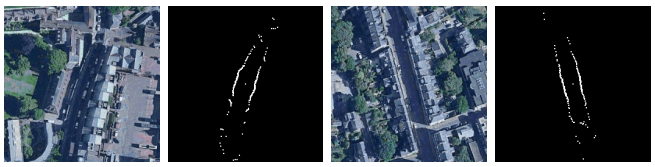


Fig. 9: Satellite images I^S (1 and 3) and the corresponding pseudo point-clouds $\mathbf{P}^S$ (2 and 4) on narrow, straight roads show a strong symmetry.

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

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

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, °)	$\pm25, \pm25, \pm180$			$\pm10, \pm10, \pm180$			$\pm25, \pm25, \pm90$			$\pm10, \pm10, \pm90$			$\pm25, \pm25, \pm45$			$\pm25, \pm25, \pm22.5$		
	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, °)	$\pm25, \pm25, \pm180$			$\pm10, \pm10, \pm180$			$\pm25, \pm25, \pm90$			$\pm10, \pm10, \pm90$			$\pm25, \pm25, \pm45$			$\pm25, \pm25, \pm22.5$		
	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, °)	$\pm25, \pm25, \pm180$			$\pm10, \pm10, \pm180$			$\pm25, \pm25, \pm90$			$\pm10, \pm10, \pm90$			$\pm25, \pm25, \pm45$			$\pm25, \pm25, \pm22.5$		
	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.

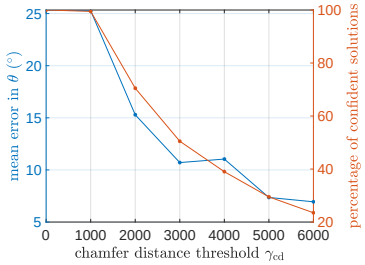


Fig. 10: Percentage of solutions considered "confident" and the mean θ error of confident solutions, for various values of γ_{cd}, evaluated on RobotCar.

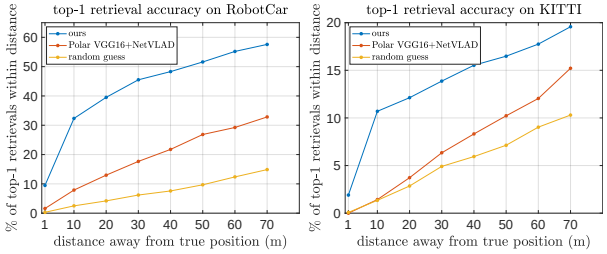


Fig. 11: Percentage of top-1 retrievals falling within a certain distance away from the true position.

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.

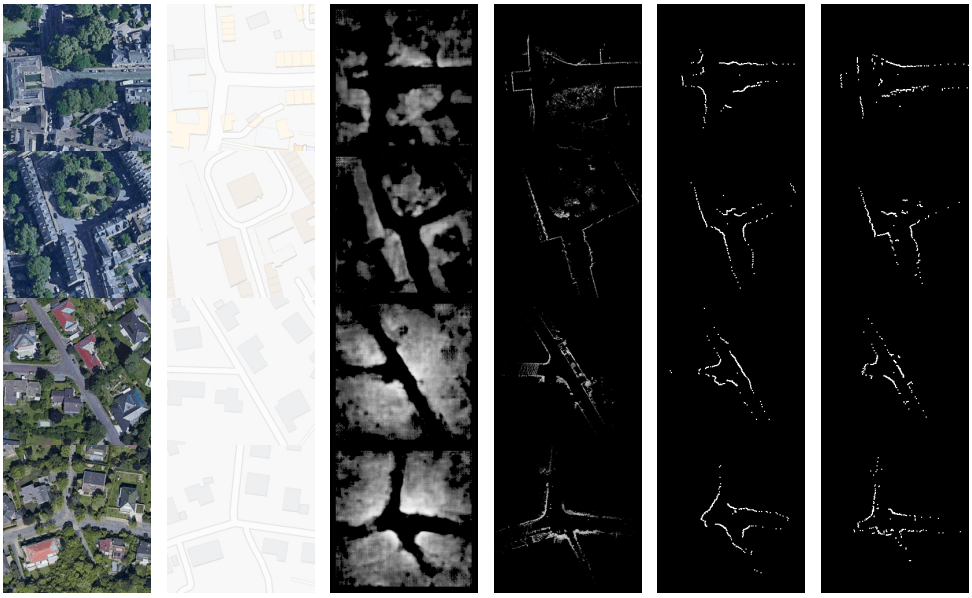
On the RobotCar Dataset, which features a city environment, our method achieves a high accuracy, having over half of all top-1 retrievals falling within 50 m of the true position. The test set for KITTI, featuring a residential area, is much more challenging as residential areas have more structurally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



(a) (b) (c) (d) (e) (f)

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.

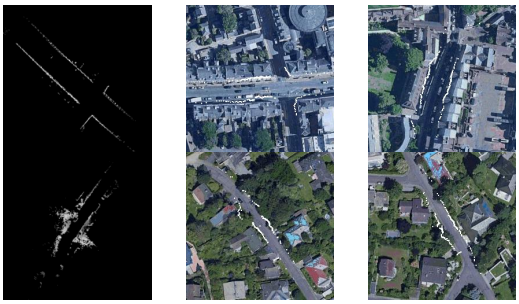


Fig. 13: Failure cases: a lidar image I^L_j along the test route (left), the satellite image and the associated pseudo point-cloud corresponding to the true position (centre), and the falsely retrieved satellite image and its associated pseudo point-cloud (right), evaluated on RobotCar (top) and KITTI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

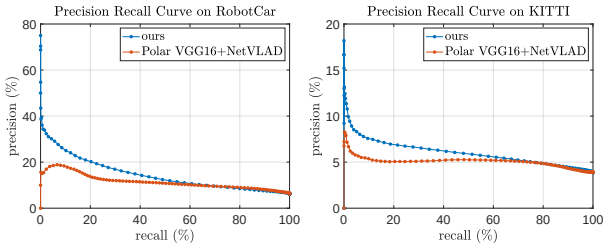


Fig. 14: Precision and recall curve for various descriptor distance thresholds.

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].

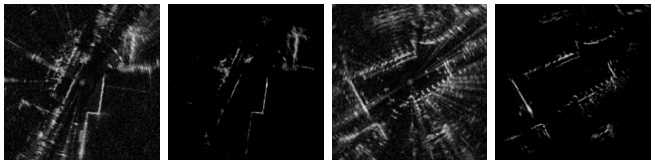


Fig. 15: Radar images (1 and 3) transformed into the appearance of lidar images (2 and 4) using the Cyclegan implementation in [47].

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.

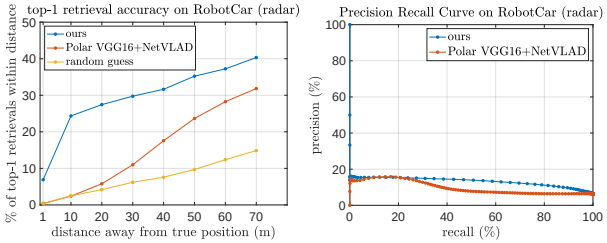


Fig. 16: Place recognition results using radar.

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.

References

[1] Sarah H Cen and Paul Newman. "Precise Ego-Motion Estimation with Millimeter-Wave Radar under Diverse and Challenging Conditions". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 6045–6052.

[2] Martin Magnusson et al. "Appearance-Based Loop Detection from 3D Laser Data Using the Normal Distributions Transform". In: *2009 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2009, pp. 23–28.

[3] Michael Bosse and Robert Zlot. "Place Recognition Using Regional Point Descriptors for 3D Mapping". In: *Field and Service Robotics*. Springer. 2010, pp. 195–204.

[4] Michael Bosse and Robert Zlot. "Place Recognition Using Keypoint Voting in Large 3D Lidar Datasets". In: *2013 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, pp. 2677–2684.

[5] Bastian Steder et al. "Place Recognition in 3D Scans Using a Combination of Bag of Words and Point Feature Based Relative Pose Estimation". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2011, pp. 1249–1255.

[6] Bastian Steder et al. "Point Feature Extraction on 3D Range Scans Taking into Account Object Boundaries". In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*. 2011, pp. 2601–2608.

[7] Li He, Xiaolong Wang, and Hong Zhang. "M2DP: A novel 3D Point Cloud Descriptor and Its Application in Loop Closure Detection". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 231–237.

[8] Jiadong Guo et al. "Local Descriptor for Robust Place Recognition Using Lidar Intensity". In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1470–1477.

[9] Federico Tombari, Samuele Salti, and Luigi Di Stefano. "A Combined Texture-Shape Descriptor for Enhanced 3D Feature Matching". In: *2011 18th IEEE International Conference on Image Processing (ICIP)*. 2011, pp. 809–812.

[10] Renaud Dubé et al. "Segmatch: Segment Based Place Recognition in 3d Point Clouds". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 5266–5272.

[11] 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)*. IEEE. 2018, pp. 4802–4809.

[12] Giseop Kim, Sunwook Choi, and Ayoung Kim. "Scan Context++: Structural Place Recognition Robust to Rotation and Lateral Variations in Urban Environments". In: *IEEE Transactions on Robotics* (2021).

[13] Lun Luo et al. "BVMatch: LiDAR-Based Place Recognition Using Bird's-Eye View Images". In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 6076–6083.

[14] Jesse Levinson, Michael Montemerlo, and Sebastian Thrun. "Map-Based Precision Vehicle Localization in Urban Environments." In: *Robotics: Science and Systems (RSS)*. Vol. 4. Atlanta, GA, USA. 2007, p. 1.

[15] Jesse Levinson and Sebastian Thrun. "Robust Vehicle Localization in Urban Environments Using Probabilistic Maps". In: *2010 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2010, pp. 4372–4378.

[16] Philipp Krüsi et al. "Lighting-Invariant Adaptive Route Following Using Iterative Closest Point Matching". In: *Journal of Field Robotics* 32.4 (2015), pp. 534–564.

[17] Nora Ripperda and Claus Brenner. "Marker-Free Registration of Terrestrial Laser Scans Using the Normal Distribution Transform". In: *Proceedings of the ISPRS working group* 4 (2005).

[18] Cihan Ulaş and Hakan Temeltaş. "3D Multi-Layered Normal Distribution Transform for Fast and Long Range Scan Matching". In: *Journal of Intelligent & Robotic Systems* 71.1 (2013), pp. 85–108.

[19] 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 (ICRA)*. IEEE. 2012, pp. 2611–2617.

[20] Will Maddern, Geoffrey Pascoe, and Paul Newman. "Leveraging Experience for Large-Scale LIDAR Localisation in Changing Cities". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 1684–1691.

[21] Timothy D Barfoot et al. "Into Darkness: Visual Navigation Based on a Lidar-Intensity-Image Pipeline". In: *Robotics Research*. Springer, 2016, pp. 487–504.

[22] Herbert Bay et al. "Speeded-Up Robust Features (SURF)". In: *Computer Vision and Image Understanding* 110.3 (2008), pp. 346–359.

[23] Zhuang Jie Chong et al. "Synthetic 2D LIDAR for Precise Vehicle Localization in 3D Urban Environment". In: *2013 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, pp. 1554–1559.

[24] Rainer Kümmerle et al. "Monte Carlo Localization in Outdoor Terrains Using Multilevel Surface Maps". In: *Journal of Field Robotics* 25.6-7 (2008), pp. 346–359.

[25] 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)*. IEEE. 2015, pp. 2814–2821.

[26] Ryan W Wolcott and Ryan M Eustice. "Robust LIDAR Localization Using Multiresolution Gaussian Mixture Maps for Autonomous Driving". In: *The International Journal of Robotics Research* 36.3 (2017), pp. 292–319.

[27] Jun-Hyuck Im, Sung-Hyuck Im, and Gyu-In Jee. "Extended Line Map-based Precise Vehicle Localization Using 3D LIDAR". In: *Sensors* 18.10 (2018), p. 3179.

[28] Xieyuanli Chen et al. "Range Image-based LiDAR Localization for Autonomous Vehicles". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 5802–5808.

[29] Fernando Caballero and Luis Merino. "DLL: Direct LIDAR Localization. A Map-based Localization Approach for Aerial Robots". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 5491–5498.

[30] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. "Generalized-ICP". In: *Robotics: Science and Systems (RSS)*. Vol. 2. 4. Seattle, WA. 2009, p. 435.

[31] Ji Zhang and Sanjiv Singh. "LOAM: Lidar Odometry and Mapping in Real-Time". In: *Robotics: Science and Systems (RSS)*. Vol. 2. 9. Berkeley, CA. 2014, pp. 1–9.

[32] Jean-Emmanuel Deschaud. "IMLS-SLAM: Scan-to-Model Matching Based on 3D Data". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2480–2485.

[33] Richard A Newcombe et al. "KinectFusion: Real-time Dense Surface Mapping and Tracking". In: *2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. Ieee. 2011, pp. 127–136.

[34] Jens Behley and Cyrill Stachniss. "Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments". In: *Robotics: Science and Systems (RSS)*. Vol. 2018. 2018, p. 59.

[35] Xieyuanli Chen et al. "SuMa++: Efficient Lidar-based Semantic Slam". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 4530–4537.

[36] Kenji Koide et al. "Globally Consistent 3D LiDAR Mapping with GPU-Accelerated GICP Matching Cost Factors". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 8591–8598.

[37] Pierre Dellenbach et al. "CT-ICP: Real-time Elastic LiDAR Odometry with Loop Closure". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 5580–5586.

[38] Charles R Qi et al. "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 652–660.

[39] Zhe Liu et al. "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 (CVPR)*. 2019, pp. 2831–2840.

[40] Kavisha Vidanapathirana et al. "LoGG3D-Net: Locally Guided Global Descriptor Learning for 3D place Recognition". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2215–2221.

[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 (CVPR)*. 2018, pp. 4470–4479.

[42] 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 (CVPR)*. 2019, pp. 12436–12445.

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

[44] Relja Arandjelovic et al. “NetVLAD: CNN Architecture for Weakly Supervised Place Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 5297–5307.

[45] Yunge Cui et al. “BoW3D: Bag of Words for Real-Time Loop Closing in 3D LiDAR SLAM”. In: *IEEE Robotics and Automation Letters* (2022).

[46] Louis Wiesmann et al. “Retriever: Point Cloud Retrieval in Compressed 3D Maps”. In: *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*. 2022.

[47] Hugues Thomas et al. “KPConv: Flexible and Deformable Convolution for Point Clouds”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6411–6420.

[48] Andrew Jaegle et al. “Perceiver: General Perception with Iterative Attention”. In: *International Conference on Machine Learning (ICML)*. PMLR. 2021, pp. 4651–4664.

[49] Li Sun et al. “Localising Faster: Efficient and Precise Lidar-Based Robot Localisation in Large-Scale Environments”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 4386–4392.

[50] Alex Kendall, Matthew Grimes, and Roberto Cipolla. “PoseNet: A Convolutional Network for Real-Time 6-DoF Camera Relocalization”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 2938–2946.

[51] Peng Yin et al. “Stabilize An Unsupervised Feature Learning for LiDAR-Based Place Recognition”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1162–1167.

[52] Jacek Komorowski. “Minkloc3D: Point Cloud Based Large-Scale Place Recognition”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2021, pp. 1790–1799.

[53] Filip Radenović, Giorgos Tolias, and Ondřej Chum. “Fine-Tuning CNN Image Retrieval with No Human Annotation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.7 (2018), pp. 1655–1668.

[54] Kamil Żywanowski et al. “MinkLoc3D-SI: 3D LiDAR Place Recognition with Sparse Convolutions, Spherical Coordinates, and Tntensity”. In: *IEEE Robotics and Automation Letters* 7.2 (2021), pp. 1079–1086.

[55] Yan Xia et al. “PVT3D: Point Voxel Transformers for Place Recognition from Sparse Lidar Scans”. In: *arXiv preprint arXiv:2211.12542* (2022).

[56] Xieyuanli Chen et al. “OverlapNet: Loop Closing for LiDAR-based SLAM”. In: *Proceedings of Robotics: Science and Systems (RSS)*. Corvalis, Oregon, USA, 2020.

[57] Junyi Ma et al. “OverlapTransformer: An Efficient and Yaw-Angle-Invariant Transformer Network for LiDAR-Based Place Recognition”. In: *IEEE Robotics and Automation Letters* (2022).

[58] Ashish Vaswani et al. “Attention is All you Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*. 2017, pp. 6000–6010.

[59] Junyi Ma et al. “SeqOT: A Spatial-Temporal Transformer Network for Place Recognition Using Sequential LiDAR Data”. In: *IEEE Transactions on Industrial Electronics* (2022).

[60] Dan Barnes et al. “The Oxford Radar Robotcar dataset: A Radar Extension to the Oxford Robotcar Dataset”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 6433–6438.

[61] Giseop Kim et al. “MulRan: Multimodal Range Dataset for Urban Place Recognition”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 6246–6253.

[62] Marcel Sheeny et al. “RADIATE: A Radar Dataset for Automotive Perception in Bad Weather”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 1–7.

[63] Keenan Burnett et al. “Boreas: A Multi-Season Autonomous Driving Dataset”. In: *arXiv preprint arXiv:2203.10168* (2022).

[64] Sarah H Cen and Paul Newman. “Radar-Only Ego-Motion Estimation in Difficult Settings via Graph Matching”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 298–304.

[65] Pou-Chun Kung, Chieh-Chih Wang, and Wen-Chieh Lin. “A Normal Distribution Transform-Based Radar Odometry Designed for Scanning and Automotive Radars”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 14417–14423.

[66] Daniel Adolfsson et al. “CFEAR Radar Odometry – Conservative Filtering for Efficient and Accurate Radar Odometry”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 5462–5469.

[67] Ziyang Hong, Yvan Petillot, and Sen Wang. “RadarSLAM: Radar Based Large-Scale SLAM in All Weathers”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 5164–5170.

[68] Yeong Sang Park, Young-Sik Shin, and Ayoung Kim. “PhaRaO: Direct Radar Odometry Using Phase Correlation”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 2617–2623.

[69] Keenan Burnett, Angela P Schoellig, and Timothy D Barfoot. “Do We Need to Compensate for Motion Distortion and Doppler Effects in Spinning Radar Navigation?” In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 771–778.

[70] Roberto Aldera et al. “What Goes Around: Leveraging a Constant-Curvature Motion Constraint in Radar Odometry”. In: *IEEE Robotics and Automation Letters* 7.3 (2022), pp. 7865–7872.

[71] Hyungtae Lim et al. “ORORA: Outlier-Robust Radar Odometry”. In: *arXiv preprint arXiv:2303.01876* (2023).

[72] Rongxi Zhang et al. “Scan Denoising and Normal Distribution Transform for Accurate Radar Odometry and Positioning”. In: *IEEE Robotics and Automation Letters* (2023).

[73] Roberto Aldera et al. “Fast Radar Motion Estimation with a Learnt Focus of Attention Using Weak Supervision”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 1190–1196.

[74] Dan Barnes and Ingmar Posner. “Under the Radar: Learning to Predict Robust Keypoints for Odometry Estimation and Metric Localisation in Radar”. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Paris, 2020. URL: https://arxiv.org/abs/2001.10789.

[75] Keenan Burnett et al. “Radar Odometry Combining Probabilistic Estimation and Unsupervised Feature Learning”. In: *Proceedings of Robotics: Science and Systems (RSS)*. Virtual, 2021.

[76] Dan Barnes, Rob Weston, and Ingmar Posner. “Masking by Moving: Learning Distraction-Free Radar Odometry from Pose Information”. In: *Conference on Robot Learning (CoRL)*. PMLR. 2020, pp. 303–316.

[77] Rob Weston et al. “Fast-MbyM: Leveraging Translational Invariance of the Fourier Transform for Efficient and Accurate Radar Odometry”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2186–2192.

[78] Ştefan Săftescu et al. “Kidnapped Radar: Topological Radar Localisation Using Rotationally-Invariant Metric Learning”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 4358–4364.

[79] Matthew Gadd, Daniele De Martini, and Paul Newman. “Look Around You: Sequence-based Radar Place Recognition with Learned Rotational Invariance”. In: *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. IEEE. 2020, pp. 270–276.

[80] Daniele De Martini, Matthew Gadd, and Paul Newman. “kRadar++: Coarse-to-Fine FMCW Scanning Radar Localisation”. In: *Sensors* 20.21 (2020). URL: https://www.mdpi.com/1424-8220/20/21/6002.

[81] Matthew Gadd, Daniele De Martini, and Paul Newman. “Contrastive Learning for Unsupervised Radar Place Recognition”. In: *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE. 2021, pp. 344–349.

[82] Jacek Komorowski, Monika Wysoczanska, and Tomasz Trzcinski. “Large-Scale Topological Radar Localization Using Learned Descriptors”. In: *International Conference on Neural Information Processing (ICONIP)*. Springer. 2021, pp. 451–462.

[83] Tsung-Yi Lin, Serge Belongie, and James Hays. “Cross-View Image Geolocalization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013, pp. 891–898.

[84] Sixing Hu et al. “CVM-Net: Cross-View Matching Network for Image-Based Ground-to-Aerial Geo-Localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 7258–7267.

[85] Liu Liu and Hongdong Li. "Lending Orientation to Neural Networks for Cross-View Geo-Localization". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 5624–5633.

[86] Yujiao Shi et al. "Where Am I Looking At? Joint Location and Orientation Estimation by Cross-View Matching". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 4064–4072.

[87] Yujiao Shi et al. "Spatial-Aware Feature Aggregation for Image Based Cross-View Geo-Localization". In: *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).

[88] Yujiao Shi et al. "Optimal Feature Transport for Cross-View Image Geo-:ocalization". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 11990–11997.

[89] Ryan S Kaminsky et al. "Alignment of 3D Point Clouds to Overhead Images". In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. IEEE. 2009, pp. 63–70.

[90] Lucas de Paula Veronese et al. "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)*. IEEE. 2015, pp. 4285–4290.

[91] Rainer Kümmerle et al. "Large Scale Graph-based SLAM using Aerial Images as Prior Information". In: *Autonomous Robots* 30.1 (2011), pp. 25–39.

[92] Can Ulas Dogruer, A Bugra Koku, and Melik Dolen. "Outdoor Mapping and Localization Using Satellite Images". In: *Robotica* 28.7 (2010), pp. 1001–1012.

[93] Marwan Hussein et al. "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 (IROS)*. IEEE. 2013, pp. 1432–1437.

[94] Minzhao Zhu et al. "AGCV-LOAM: Air-Ground Cross-View Based LiDAR Odometry and Mapping". In: *2020 Chinese Control And Decision Conference (CCDC)*. IEEE. 2020, pp. 5261–5266.

[95] Mengyin Fu et al. "LiDAR-based Vehicle Localization on the Satellite Image via a Neural Network". In: *Robotics and Autonomous Systems* 129 (2020), p. 103519.

[96] Florian Fervers et al. "Continuous Self-Localization on Aerial Images Using Visual and Lidar Sensors". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 7028–7035.

[97] Jonghwi Kim, Yonghoon Cho, and Jinwhan Kim. "Urban Localization Based on Aerial Imagery by Correcting Projection Distortion". In: *Autonomous Robots* (2022), pp. 1–14.

[98] Ryan W Wolcott and Ryan M Eustice. "Visual Localization within Lidar Maps for Automated Urban Driving". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2014, pp. 176–183.

[99] Tim Caselitz et al. "Monocular Camera Localization in 3D Lidar Maps". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 1926–1931.

[100] Yuquan Xu et al. “3D Point Cloud Map Based Vehicle Localization Using Stereo Camera”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2017, pp. 487–492.

[101] 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 (ICRA)*. IEEE. 2010, pp. 881–886.

[102] Xipeng Wang, Steve Vozar, and Edwin Olson. “FLAG: Feature-based Localization between Air and Ground”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 3178–3184.

[103] Federico Boniardi et al. “Robust LiDAR-based Localization in Architectural Floor Plans”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 3318–3324.

[104] Malcolm Mielle, Martin Magnusson, and Achim J Lilienthal. “The Auto-Complete Graph: Merging and Mutual Correction of Sensor and Prior Maps for SLAM”. In: *Robotics* 8.2 (2019), p. 40.

[105] Peter A Iannucci, Lakshay Narula, and Todd E Humphreys. “Cross-Modal Localization: Using Automotive Radar for Absolute Geolocation within a Map Produced with Visible-Light Imagery”. In: *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. IEEE. 2020, pp. 285–296.

[106] David G Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* 60 (2004), pp. 91–110.

[107] Huan Yin et al. “Radar Style Transfer for Metric Robot Localisation on Lidar Maps”. In: *CAAI Transactions on Intelligence Technology* (2022).

[108] Somi Jeong et al. “Learning to Find Unpaired Cross-Spectral Correspondences”. In: *IEEE Transactions on Image Processing* 28.11 (2019), pp. 5394–5406.

[109] 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 (CVPR)*. 2013, pp. 3057–3064.

[110] Georgios Floros, Benito Van Der Zander, and Bastian Leibe. “OpenStreetSLAM: Global Vehicle Localization Using OpenStreetMaps”. In: *2013 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, pp. 1054–1059.

[111] Olga Vysotska and Cyrill Stachniss. “Improving SLAM by Exploiting Building Information from Publicly Available Maps and Localization Priors”. In: *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science* 85 (2017), pp. 53–65.

[112] 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)*. IEEE. 2018, pp. 6341–6348.

[113] Fan Yan, Olga Vysotska, and Cyrill Stachniss. “Global Localization on OpenStreetMap Using 4-bit Semantic Descriptors”. In: *2019 European Conference on Mobile Robots (ECMR)*. IEEE. 2019, pp. 1–7.

[114] Younghun Cho et al. “OpenStreetMap-based LiDAR Global Localization in Urban Environment without a Prior LiDAR Map”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 4999–5006.

[115] Liz Murphy and Paul Newman. “Planning Most-Likely Paths from Overhead Imagery”. In: *2010 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2010, pp. 3059–3064.

[116] Luís Carlos Santos et al. “Occupancy Grid and Topological Maps Extraction from Satellite Images for Path Planning in Agricultural Robots”. In: *Robotics* 9.4 (2020), p. 77.

[117] Lorenzo Nardi and Cyrill Stachniss. “Uncertainty-Aware Path Planning for Navigation on Road Networks Using Augmented MDPs”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 5780–5786.

[118] Ian Goodfellow et al. “Generative Adversarial Networks”. In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[119] Phillip Isola et al. “Image-to-Image Translation with Conditional Adversarial Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (ICCV)*. 2017, pp. 1125–1134.

[120] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2223–2232.

[121] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. “Unsupervised Image-to-Image Translation Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017, pp. 700–708.

[122] Xun Huang et al. “Multimodal Unsupervised Image-to-Image Translation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 172–189.

[123] Taesung Park et al. “Contrastive Learning for Unpaired Image-to-Image Translation”. In: *European Conference on Computer Vision (ECCV)*. Springer. 2020, pp. 319–345.

[124] Amjad Almahairi et al. “Augmented CycleGAN: Learning Many-to-Many Mappings from Unpaired Data”. In: *International Conference on Machine Learning (ICML)*. PMLR. 2018, pp. 195–204.

[125] Hsin-Ying Lee et al. “Diverse Image-to-Image Translation via Disentangled Representations”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 35–51.

[126] Zhixin Shu et al. “Deforming Autoencoders: Unsupervised Disentangling of Shape and Appearance”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 650–665.

[127] Wayne Wu et al. “TransGaGa: Geometry-Aware Unsupervised Image-to-Image Translation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 8012–8021.

[128] Xianglei Xing et al. “Unsupervised Disentangling of Appearance and Geometry by Deformable Generator Network”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 10354–10363.

[129] Ioan Andrei Barsan et al. “Learning to Localize Using a LiDAR Intensity Map”. In: *Conference on Robot Learning (CoRL)*. PMLR. 2018, pp. 605–616.

[130] Paul J Besl and Neil D McKay. “Method for Registration of 3-D Shapes”. In: *Sensor fusion IV: Control Paradigms and Data Structures*. Vol. 1611. International Society for Optics and Photonics. 1992, pp. 586–606.

[131] 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)*. 2018, pp. 607–623.

[132] Charles Ruizhongtai Qi et al. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).

[133] Yue Wang et al. “Dynamic Graph CNN for Learning on Point Clouds”. In: *ACM Transactions On Graphics (tog)* 38.5 (2019), pp. 1–12.

[134] Yue Wang and Justin M Solomon. “PRNet: Self-Supervised Learning for Partial-to-Partial Registration”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).

[135] Wentao Yuan et al. “DeepGMR: Learning Latent Gaussian Mixture Models for Registration”. In: *European Conference on Computer Vision (ECCV)*. Springer. 2020, pp. 733–750.

[136] Andy Zeng et al. “3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1802–1811.

[137] Kin Leong Ho and Paul Newman. “Detecting Loop Closure with Scene Sequences”. In: *International Journal of Computer Vision* 74.3 (2007), pp. 261–286.

[138] Temple F Smith, Michael S Waterman, et al. “Identification of Common Molecular Subsequences”. In: *Journal of Molecular Biology* 147.1 (1981), pp. 195–197.

[139] Yang Liu and Hong Zhang. “Visual Loop Closure Detection with a Compact Image Descriptor”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2012, pp. 1051–1056.

[140] Michael J Milford and Gordon F Wyeth. “SeqSLAM: Visual Route-Based Navigation for Sunny Summer Days and Stormy Winter Nights”. In: *2012 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2012, pp. 1643–1649.

[141] Sayem Mohammad Siam and Hong Zhang. “Fast-SeqSLAM: A Fast Appearance Based Place Recognition Algorithm”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 5702–5708.

[142] Peter Hansen and Brett Browning. “Visual Place Recognition using HMM Sequence Matching”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2014, pp. 4549–4555.

[143] Olga Vysotska and Cyrill Stachniss. “Lazy Data Association for Image Sequences Matching Under Substantial Appearance Changes”. In: *IEEE Robotics and Automation Letters* 1.1 (2015), pp. 213–220.

[144] Pierre Sermanet et al. "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks". In: *2nd International Conference on Learning Representations (ICLR), Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2014. URL: http://arxiv.org/abs/1312.6229.

[145] Olga Vysotska and Cyrill Stachniss. "Effective Visual Place Recognition Using Multi-Sequence Maps". In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1730–1736.

[146] Olga Vysotska. "Visual Place Recognition in Changing Environments". PhD thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2019. URL: https://hdl.handle.net/20.500.11811/8010.

[147] Stefan Schubert, Peer Neubert, and Peter Protzel. "Unsupervised Learning Methods for Visual Place Recognition in Discretely and Continuously Changing Environments". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 4372–4378.

[148] Sourav Garg and Michael Milford. "SeqNet: Learning Descriptors for Sequence-based Hierarchical Place Recognition". In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 4305–4312.