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.

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, 37]. 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. Saffescu 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 $(x, y, \theta)$. 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 a map (e.g., satellite) image and a live data image as inputs. For example, [48] 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 $A \in A$ and live data image $B \in B$ 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 $[x \ y \ \theta]^T$, such that by rotating $B$ by $\theta$, followed by a translation of $[x \ y]^T$, one can pixel-wise align $B$ onto $A$. The image generation can be formulated as:

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

where $\alpha = [x \ y]^T$. $\tilde{B}_{\theta, \alpha}$ is a generated image of modality $B$ that synthesises the input live sensor image $B$ applied with a rotation of $\theta$, followed by a translation of $\alpha = [x \ y]^T$. 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:

$$f_R(A, B) \rightarrow B_{\theta}$$  \hspace{1cm} (2)$$
$$f_C(A, B_{\theta}) \rightarrow \tilde{B}_{\theta, \alpha}$$  \hspace{1cm} (3)$$

Here $f_R$ is a function that infers the rotation offset $\theta$ between $A$ and $B$, and outputs $B_{\theta}$, which is input image $B$ rotated by $\theta$. Now, $B_{\theta}$ is rotation-aligned with the map frame, and therefore offset with $A$ only by a translation, which CNNs can naturally handle. $f_C$ 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 $A$ is rotated a number of times with small increments to form a stack of rotated images 

$\{B\} = \{B_{\theta_1}, B_{\theta_2}, \ldots, B_{\theta_n}\}$, 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 $B_{\theta}$. 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.](image)

If metrically accurate heading ground truth $\theta$ is available, then one can rotate $B$ to form an image target to $B_{\theta}$ used for supervising the rotation inference, as in Figure 5. 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 $B_{\theta}$, $f_R$ should also be able to output $A_{\theta}$ from a stack of rotated map images $\{A\}$, where $A_{\theta}$ is rotation-aligned with $B_{\theta}$. Specifically, if we have $B_{\theta} = B_{\theta}$, then the softmaxed map image from $\{A\}$ should be $A$, as $A$ and $B$ 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 $B_{\theta}$, $B_{\theta}$ is then used as input to the second pass through network $f_R$, together with a stack of map images $\{A\} = \{A, A_{\theta_1}, A_{\theta_2}, \ldots, A_{\theta_n}\}$. The rotation angles $[\phi_0, \phi_1, \ldots, \phi_m]$ 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 $B_{\theta}$ 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 $B_{\theta}$ to be $B_{\theta}$, as $B_{\theta}$ is rotation-aligned with $A$. Our approach is shown in Figure 4. We use an increment of $2^\circ$ 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 $B_0$ we seek to generate a synthetic image $\hat{B}_{\theta,a}$ as in (3), where $\hat{B}_{\theta,a}$ is pixel-wise aligned with $A$. (39) learns the image generation function by a supervised approach, concatenating $A$ and $B_0$, and applying an encoder-decoder architecture, as shown in Figure 5. This is possible since a target for the synthetic image $\hat{B}_{\theta,a}$ 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 $B$ 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 $B_2^\gamma$ 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. $B_2^\gamma$ and $B_2^\gamma$ 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 $\hat{B}_1^\gamma$ that is $B_1$ shifted by a translation $\gamma$. 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 $\gamma$ is known, thus we can shift $B_1^\gamma$ by $\gamma$ to produce the target $B_1^\gamma$.

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 $E_p^\gamma$ that discovers the translation offset between an image of modality $A$ and another of $B$. Taking $A$ and $B_{\theta}$ as inputs, $E_p^\gamma$ should encode the unknown translation offset $\alpha$ between them. Concurrently, $B_{\theta}$ is encoded by $E_a$, where the latent space is combined with the latent space produced...
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 \( B_\theta \) by an unknown translation \( \beta \). Using the same encoder-decoder combination as before, we can take \( A' \) and \( B_\theta \) to generate a synthetic image \( \tilde{B}_{\theta, \beta} \). Furthermore, given \( B_\theta \) 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 \( B_\theta \) by \( \beta - \alpha \) to get \( \tilde{B}_{\theta, \beta - \alpha} \). Using \( \tilde{B}_{\theta, \beta - \alpha} \) and \( B_\theta \), we can generate \( \tilde{B}_{\theta, \beta} \) 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 \( \tilde{B}_{\theta, \beta - \alpha} \) 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 \( B_\theta \). Second, \( \tilde{B}_{\theta, \alpha} \) and \( \tilde{B}_{\theta, \beta} \) must have the correct translations \( \alpha \) and \( \beta \), 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 \( B_\theta \), 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 \( \hat{\alpha} = [\hat{x}, \hat{y}]^T \), 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 \( \alpha \), 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 \( \tilde{B}_{\theta, \beta} \) and \( B_\theta \), the translation offset by correlation maximisation is found to be \( \hat{\beta} \). If we replace \( \tilde{B}_{\theta, \beta} \) with \( B_{\theta, \alpha} \) 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
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_drive0027, 20110930_drive0028, and 20110930_drive0029. Sequence 20110926_drive0117 is used for validation. Finally, data in 20110930_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 plane using Bing Maps Platform [1], as high-definition Google satellite imagery is unavailable at the place of interest.

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

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.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Mean Error (metric)</th>
<th>Mean Error (pixel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RobotCar (ours)</td>
<td>3.44 5.40 3.03</td>
<td>3.97 6.23</td>
</tr>
<tr>
<td>MulRan (ours)</td>
<td>6.02 7.02 2.92</td>
<td>7.64 8.91</td>
</tr>
<tr>
<td>RobotCar (RSL-Net [39], supervised)</td>
<td>2.74 4.26 3.12</td>
<td>3.16 4.92</td>
</tr>
<tr>
<td>MulRan (RSL-Net)</td>
<td>5.85 7.11 1.88</td>
<td>7.42 9.03</td>
</tr>
</tbody>
</table>

**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.
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.

![Fig. 8](image)

**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 \( \{B_{\theta A}, \ldots, B_{\theta A} \} \), from which \( f \theta \) discovers \( B_{\theta} \) that is \( B \) rotated to be rotation-aligned with \( A \). This process also infers the heading estimate \( \hat{\theta} \). \( A \) and \( B_{\theta} \) are used to generate a synthetic image \( \tilde{B}_{\theta,\alpha} \), that has the same appearance and observed scene as \( B_{\theta} \) and is pose-aligned with \( A \). \( \tilde{B}_{\theta,\alpha} \) and \( B_{\theta} \) are projected to deep embeddings \( \tilde{B}_{\theta,\alpha} \) and \( B_{\theta} \), where the estimate for the translation offset \( \hat{\alpha} \) is found by correlation maximisation.

![Fig. 9](image)

**Fig. 9:** Training (blue), validation (green), and test (red) trajectories for RobotCar (top left), KAIST (top right), Sejong (bottom left) and 2011003_drive0034 (bottom right). Certain data are removed to avoid overlap between the splits.

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.

**TABLE II:** Mean error for lidar localisation against satellite imagery.

<table>
<thead>
<tr>
<th></th>
<th>Mean Error (metric)</th>
<th>Mean Error (pixel)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( x (\text{m}) )</td>
<td>( y (\text{m}) )</td>
</tr>
<tr>
<td>RobotCar (ours)</td>
<td>1.54</td>
<td>1.85</td>
</tr>
<tr>
<td>KITTI (ours)</td>
<td>3.05</td>
<td>3.13</td>
</tr>
<tr>
<td>RobotCar (RSL-Net)</td>
<td>2.31</td>
<td>2.55</td>
</tr>
<tr>
<td>KITTI (RSL-Net)</td>
<td>2.45</td>
<td>2.79</td>
</tr>
</tbody>
</table>

**C. 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 \( A \) 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 \( B_{\theta} \). We can apply a known small translation offset \( \delta \) to \( A \) to form \( A_{\delta} \). Taking \( A_{\delta} \) and \( B_{\theta} \), we can generate \( \tilde{B}_{\theta,\alpha+\delta} \). Finally, we can compute a translation offset \( \delta \) by passing \( B_{\theta,\alpha+\delta} \) and \( B_{\theta,\alpha} \) through the learned embeddings and maximising correlation.

Let \( d_{\text{intro}} = \| \delta - \hat{\delta} \| \). A large value of \( d_{\text{intro}} \) 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 $d_{\text{intro}}$ 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 $\delta$ to be $[10 \ 10]^T$, and $d_{\text{intro}}$ 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 onboard 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


