Radar Odometry Combining Probabilistic Estimation and Unsupervised Feature Learning

Keenan Burnett*, David J. Yoon*, Angela P. Schoellig, and Timothy D. Barfoot
Institute for Aerospace Studies, University of Toronto

Abstract—This paper presents a radar odometry method that combines probabilistic trajectory estimation and deep learned features without needing groundtruth pose information. The feature network is trained unsupervised, using only the on-board radar data. With its theoretical foundation based on a data likelihood objective, our method leverages a deep network for processing rich radar data, and a non-differentiable classic estimator for probabilistic inference. We provide extensive experimental results on both the publicly available Oxford Radar RobotCar Dataset and an additional 100 km of driving collected in an urban setting. Our sliding-window implementation of radar odometry outperforms most hand-crafted methods and approaches the current state of the art without requiring a groundtruth trajectory for training. We also demonstrate the effectiveness of radar odometry under adverse weather conditions. Code for this project can be found at: [https://github.com/utiasASRL/hero_radarmodule_odometry](https://github.com/utiasASRL/hero_radarmodule_odometry)

I. INTRODUCTION

While the reliability of autonomous vehicles continues to improve, operating in rain and snow remains a challenge. Most autonomous driving systems rely primarily on a combination of cameras and lidar for perception, with radar sensors taking a back-seat role [13]. Cen and Newman [15] presented a successful application of a scanning radar sensor to large-scale outdoor ego-motion estimation. Their work has inspired a resurgence of research into radar-based perception and navigation systems. Compared to lidar, radar is more robust to precipitation due to its longer wavelength. For this reason, radar may be a key to enabling self-driving vehicles to operate in adverse weather. The ultimate goal of this research is to approach the performance of lidar-based algorithms in ideal conditions and surpass them in adverse conditions.

Previous works in this area have made significant progress towards radar-based odometry [2, 4, 9, 11, 14, 16, 26, 34, 39] and place recognition [20, 22, 31, 43, 50]. However, previous approaches to radar odometry have either relied on handcrafted feature extraction [2, 4, 14, 16, 26, 34], or a (self-)supervised learning algorithm [9, 11] that relies on trajectory groundtruth. Barnes and Posner [9] previously showed that learned features have the potential to outperform hand-crafted features. On the other hand, it has not yet been shown whether correlative scan matching systems can be scaled up to large-scale mapping and localization. In order to address these limitations, we propose a method that is able to learn features directly from radar data without relying on groundtruth pose information.

In this work, we present an unsupervised radar odometry pipeline that approaches the state of the art as reported on the Oxford Radar RobotCar Dataset [10]. Our network parameters are trained using only the on-board radar sensor, alleviating the need for an accurate groundtruth trajectory. To our knowledge, this is the first example of a totally unsupervised radar odometry pipeline. We show additional experimental results on 100 km of radar data collected on the data-taking platform shown in Figure 1. We also provide a comparison of radar odometry performance in ideal and harsh weather conditions.

Our approach is based on the Exactly Sparse Gaussian Variational Inference (ESGVI) parameter learning framework of Barfoot et al. [8], a nonlinear batch state estimation framework that provides a family of scalable estimators from a variational objective. Model parameters can be optimized jointly with the state using a data likelihood objective. Yoon et al. [51] recently applied the framework to train a deep network, demonstrating feature learning for lidar odometry with only on-board lidar data. We extend their methodology and apply it to radar odometry. Our approach is modular, enabling the use of modern deep learning tools and classical estimators with minimal interface requirements. Importantly, our method does not require the estimator to be differentiable, a limitation of some other learning-based methods [11].

The hybridization of deep learning and probabilistic state estimation allows us to achieve the best of both worlds. Deep learning can be leveraged to process rich sensor data, while classical estimators can be used to deal with probabilities and out-of-distribution samples through outlier rejection schemes, motion priors, and other estimation tools. Furthermore, classical estimators make the incorporation of additional sensors and constraints relatively straightforward.
We review related work in Section II. An overview of ESGVI parameter learning is presented in Section III and the methodology of our radar odometry is in Section IV. Experimental results are presented in Section V. Concluding remarks and future work are in Section VI.

II. RELATED WORK

Initially, radar-based navigation was reliant on hand-tuned detectors specifically designed for radar [12, 29]. One such detector is CFAR (Constant False Alarm Rate), which uses a sliding window to find relative peaks in radar data [41]. Although CFAR works well for detecting aircraft against a blank sky, it struggles to discriminate between clutter and objects of interest on the ground where the relative differences in radar cross section are low. These early methods relied primarily on location information to match detections and were thus limited by the accuracy of their data association.

Subsequent works have sought to improve feature detection and matching. Chechin et al. [18] side-stepped this problem by applying correlative scan matching between entire radar scans via the Fourier-Mellin transform. Although correlation-based techniques have been shown to achieve excellent odometry performance, it remains challenging to apply them to large-scale mapping and localization due in part to their high data storage requirements during mapping.

Recent works have taken advantage of improvements to frequency-modulated continuous wave (FMCW) radar sensors. The radar sensor produced by Navtech [1] is popular for navigation research due to its combination of high range resolution, angular resolution, range, and $360^\circ$ field of view.

Cen and Newman [15] demonstrated large-scale outdoor ego-motion estimation with a scanning radar. Their work inspired a resurgence of research into radar-based perception and navigation systems. Research in this area has also been accelerated by the introduction of the Oxford Radar RobotCar Dataset [10]. Other datasets that feature a scanning radar include the MuLan Dataset [31] for multimodal place recognition, and the RADIATE Dataset [44] for object detection and tracking in adverse weather.

Cen and Newman [16] later provided an update to their radar odometry pipeline with an improved gradient-based feature detector along with a new graph matching strategy. Aldera et al. [3] learn an attention policy to down-sample the number of measurements provided to the data association stage, thus speeding up the odometry pipeline. Aldera et al. [4] train a classifier on the principal eigenvector of their graph matching problem in order to predict and correct for failures in radar odometry. Park et al. [49] applied the Fourier-Mellin Transform to log-polar images computed from down-sampled Cartesian images. Barnes et al. [11] demonstrated a fully differentiable, correlation-based radar odometry pipeline. Their approach learns a binary mask to remove distractor features before using brute force search to find the pose with the minimum cross correlation. Hong et al. [26] used vision-based features and graph matching to demonstrate the first radar-based SLAM system to operate in adverse weather. Burnett et al. [14] demonstrated that motion distortion and Doppler effects can have an adverse effect on radar-based navigation, but that these effects can be compensated using continuous-time estimation techniques.

These works represent significant steps towards radar-based autonomy. However, each of these methods relies on either hand-crafted feature detectors and descriptors or cumbersome phase correlation techniques. Barnes and Posner [9] showed that learned features can result in superior radar odometry performance. Their work currently represents the state of the art for point-based radar odometry. Despite these results, their approach requires the estimator to be differentiable and they require groundtruth poses as a supervisory signal. Our approach does not suffer from either of these drawbacks.

Other works have focused on radar-based place recognition. S˘aftescu et al. [43] perform place recognition by learning rotationally invariant descriptors within a metric feature space. Gadd et al. [22] improve on this work by incorporating sequence-based place recognition based on SeqSLAM [37]. De Martini et al. [20] proposed a two-stage system to integrate topological localization with metric pose estimation. A related avenue of research has been to localize radar scans using existing satellite imagery [46, 47]. Further research has tackled radar-based perception through occupancy [48], traversability [12, 49], and semantic segmentation [50].

Barfoot et al. [8] presented the ESGVI framework and showed that model parameters can be jointly optimized along with the state using an Expectation-Maximization (EM) iterative optimization scheme on a data likelihood objective. In the E-step, model parameters are held fixed, and the state is optimized. In the M-step, the state distribution is held fixed, and the model parameters are optimized. This idea originates from a line of work that applied EM for linear system identification [24, 45]. Ghahramani and Roweis [25] extended the idea to simple nonlinearities approximated with Gaussian radial basis functions. Wong et al. [50] first applied EM under the ESGVI framework to large-scale trajectory estimation by learning noise models robust to outliers. Yoon et al. [51] demonstrated the first application of ESGVI parameter learning to a deep network, where lidar features for odometry were learned without groundtruth. We apply their methodology and adapt it for radar odometry.

Our framework shares similarities with the Variational Autoencoder (VAE) [43] framework, as both start from a data likelihood objective and optimize the Evidence Lower Bound (ELBO). In contrast to the VAE, which approximates latent state inference with a network, our framework applies classic state estimation (e.g., factor graph optimization). Extensions to the VAE for problems with graphical structure exist [28], but our method makes use of existing estimation tools familiar to the robotics field. Most similar to our framework is the work of DeTone et al. [21]. They present a self-supervised visual odometry framework that has a deep network frontend trained according to a bundle adjustment backend. Compared to their framework, ours is based on a probabilistic objective and can handle uncertainty in the posterior estimates.
III. EXACTLY SPARSE GAUSSIAN VARIATIONAL INFERENCE PARAMETER LEARNING

This section summarizes parameter learning in the ESGVI framework as presented in prior work [8, 50, 51]. The loss function is the negative log-likelihood of the observed data,

\[ \mathcal{L} = -\ln p(z|\theta), \]  

where \( z \) is the data (e.g., radar measurements), and \( \theta \) are the parameters (e.g., network parameters). We apply the usual EM decomposition after introducing the latent trajectory, \( x \), which is written as

\[ \mathcal{L} = \int_{-\infty}^{\infty} q(x) \ln \left( \frac{p(x|z, \theta)}{q(x)} \right) dx - \int_{-\infty}^{\infty} q(x) \ln \left( \frac{p(x|z)}{q(x)} \right) dx, \]

where we define our posterior approximation as a multivariate Gaussian distribution, \( q(x) = \mathcal{N}(\mu, \Sigma) \).

We work with the upper bound term, commonly referred to as the (negative) ELBO. Applying the definition of entropy of a Gaussian and dropping constants, we can rewrite the term as the ESGVI loss functional,

\[ V(q|\theta) = \mathbb{E}_q[\phi(x, z|\theta)] + \frac{1}{2} \ln \left( |\Sigma^{-1}| \right), \]

where \( \mathbb{E}[\cdot] \) is the expectation operator, \( |\cdot| \) is the matrix determinant, and we define the joint factor \( \phi(x, z|\theta) = -\ln p(x, z|\theta) \).

We apply Generalized EM (GEM) [33], a variation of EM that does not run the M-step to completion (convergence), to gradually optimize the data likelihood, \( \mathcal{L} \). In the E-step, we hold \( \theta \) fixed and optimize \( V(q|\theta) \) for \( q(x) \). In the M-step, we hold \( q(x) \) fixed and optimize \( V(q|\theta) \) for \( \theta \). When the expectation over the posterior, \( q(x) \), is approximated at the mean of the Gaussian, the E-step is the familiar Maximum A Posteriori (MAP) estimator [8].

IV. UNSUPERVISED DEEP LEARNING FOR RADAR ODOMETRY

A. Problem Definition

In this subsection, we summarize the sliding-window odometry formulation presented by Yoon et al. [51]. The state we estimate at time \( t_k \) is \( x_{k} \in SE(3) \), where the pose \( T_{k,0} \in SE(3) \) is a transformation between frames at \( t_k \) and \( t_0 \), and \( \omega_k \in \mathbb{R}^6 \) is the body-centric velocity.\(^1\) We optimize a sliding-window of \( w \) frames, \( t_\tau, \ldots, t_{\tau+w-1} \), where each state has a corresponding radar scan. The first pose of the window, \( T_{\tau,0} \), is locked (not optimized) and treated as the reference frame for keypoint matching.

Our joint factor, \( \phi(x_k, z_{\ell}|\theta) \), splits into motion prior factors and measurement factors. Figure 2 shows an example factor graph illustration. We write \( \phi(x_k, z_{\ell}|\theta) \) as

\[ \sum_{k=\tau+1}^{\tau+w-1} \left\{ \phi^p(x_{k-1}, x_k) + \sum_{\ell=1}^{L_k} \phi^m(z_{\ell}^k, r_{\ell}^k|x_{\tau}, x_k, \theta) \right\}, \]

\(^1\)While the acronym stays the same, we work with the negative log-likelihood and are technically applying Expectation Minimization.

\(^2\)Despite the radar being a 2D sensor, we formulate our problem in \( SE(3) \) to be compatible with other 3D sensor modalities.
Radar Input belonging to cells with less than 5% valid pixels are rejected. Pixels in each square cell for robustness to noise. Keypoints mask of valid pixels. We then threshold on the ratio of valid intensity, where exceeding the threshold is considered valid. Projecting the result into Cartesian space produces a binary output of each encoder block before concatenation into a 248-channel tensor. In our approach, the weight score is a 3-channel tensor. The detector score tensor is then partitioned into channel tensor and the detector score is a 1-channel tensor. In our architecture, the weight score is a 3-channel tensor. In our network, the weight score is a 3-channel tensor.

B. Network

Our network is based on the architecture presented by Barnes and Posner [9], which is a U-Net [42] style convolutional encoder-multi-decoder architecture to output radar keypoints, weights, and descriptors. The input to the network is a 2D Cartesian projection of the polar radar data. Example radar scans are shown at the top of Figure 2. An illustration of our network architecture is shown in Figure 3. A dense descriptor map is created by resizing the output of each encoder block before concatenation into a 248-channel tensor. In our approach, the weight score is a 3-channel tensor and the detector score is a 1-channel tensor. The detector score tensor is then partitioned into \( N = 400 \) equally sized cells, with each producing a candidate 2D keypoint. A spatial softmax within each cell, followed by a weighted summation of pixel coordinates produces the image-space keypoint coordinates of each cell. The corresponding descriptor and weight score vectors are bilinearly sampled using the keypoint coordinates. The image coordinates are converted to metric coordinates using the known m/pixel resolution. Finally, we formulate the 3D homogenous keypoint, \( z_k \), by appending a 0 as a third coordinate, and a 1 as the fourth homogenous element.

Our detector produces a candidate keypoint for each square cell in a uniformly partitioned radar image. This leads to candidate keypoints in regions of the image that are void of data (i.e., black regions of a radar image). As we are training without ground truth, we found it necessary in practice to mask (reject) these keypoints. We threshold each azimuth of the polar radar scan by a scalar multiple, \( \beta = 3 \), of its mean intensity, where exceeding the threshold is considered valid. Projecting the result into Cartesian space produces a binary mask of valid pixels. We then threshold on the ratio of valid pixels in each square cell for robustness to noise. Keypoints belonging to cells with less than 5% valid pixels are rejected.

A dense match between each keypoint descriptor and the reference descriptor tensor is applied with a softmax to preserve differentiability. We compute the dot product between each keypoint descriptor and all descriptors of the reference:

\[
\mathbf{c}_k^T = \mathbf{d}_k^T \left[ \mathbf{d}_1^T \cdots \mathbf{d}_N^T \right],
\]

where \( \mathbf{d}_k^T \) is the descriptor vector of keypoint \( z_k \), and \( \mathbf{d}_1^T, \ldots, \mathbf{d}_N^T \) are the descriptor vectors of the reference. We apply a softmax on \( \mathbf{c}_k^T \) and compute a weighted summation. The reference match for keypoint \( z_k \) is therefore

\[
\mathbf{r}_k^T = \left[ \mathbf{p}_1^T \cdots \mathbf{p}_N^T \right] \times \text{softmax}(T\mathbf{c}_k^T),
\]

where \( T = 100 \) is a softmax temperature constant, \( \mathbf{p}_1^T, \ldots, \mathbf{p}_N^T \) are the homogeneous reference coordinates, and \( \mathbf{p}_1^T \in \mathbb{R}^4 \). The weight score vectors of each keypoint are assembled into matrices with the following decomposition [35, 51]:

\[
\mathbf{W}_k = \begin{bmatrix} \mathbf{R} & 0 \\ 0 & \mathbf{R} \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ d_1 & 1 & 0 \\ d_2 & d_3 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ d_2 & 0 & \exp(d_2) \\ d_3 & \exp(d_2) & 1 \end{bmatrix},
\]

where \((d_1, d_2, d_3)\) is the weight score vector corresponding to keypoint \( z_k \), and \( c = 10^4 \) is a constant corresponding to the (inverse) variance of the third coordinate (which will always be 0). Visualizations of the learned keypoint covariances, \( \mathbf{R}^{-1} \), are shown in Figure 4 as uncertainty ellipses.

C. Training and Inference

We follow the training methodology of Yoon et al. [51], which blends the GEM iterative optimization scheme required by ESGVI with Stochastic Gradient Descent (SGD) applied in conventional network training. The E-step, which optimizes for the current best posterior estimate, \( q(x) \), is simply included as part of the forward propagation routine.
D. Outlier Rejection

We apply the Geman-McClure robust cost function on the measurement factors, $\phi^m$, in the E-step for outlier rejection. For the M-step, we follow Yoon et al. [51] and apply a constant threshold, $\alpha$, on the squared Mahalanobis distance of the measurement factor with the current best posterior estimate, 

$$e_k^T W_k e_k > \alpha, \quad (12)$$

where $e_k$ is as defined in [5]. We do not backpropagate keypoint matches that are greater than $\alpha = 16$.

After training, we improved odometry performance by rejecting keypoints with inverse covariances that have a log determinant, $\log|\mathbf{R}|_l$, less than a threshold ($\eta = 4.0$). We also use RANSAC at test time and only estimate on the inliers.

V. EXPERIMENTAL RESULTS

A. Experiment Setup

We evaluate our approach, Hybrid-Estimate Radar Odometry (HERO), on the publicly available Oxford Radar RobotCar Dataset [10] and on our own dataset, collected using the data-taking platform in Figure 1. The Oxford dataset is divided into 32 sequences, each approximately 10 km in length. We follow existing work [9] by training on 24 sequences, validating on 1 sequence, and testing on the same 7 sequences as [9].

Our implementation is a hybrid between Python and C++, where network-related code is implemented in Python using PyTorch [40], and estimation-related code is implemented using STEAM [7] an open-source C++ estimation library. When training the network, we use a fixed random seed and the Adam optimizer [32] with a learning rate of $1 \times 10^{-5}$ and a mini-batch size of 1 window ($w = 4$) for up to 100k iterations. The Navtech radar sensor used in the Oxford dataset has a range resolution of 0.0438 m/bin. To minimize projection errors, we chose a Cartesian resolution of 0.2628 m/pixel to be an integer multiple of the radar resolution. The Cartesian radar images are made to be square with a width of 640 pixels. For the spatial softmax operation, each cell is 32 $\times$ 32 pixels, resulting in 400 total keypoints. We use a softmax temperature of 100. For the motion prior, $Q_c$ is made to be diagonal. The entries of $Q_c^{-1}$ can be thought of as penalty terms on body-centric linear and angular acceleration. It is possible to learn $Q_c$, as is done [50]. However, in our implementation, the values are hand-tuned. Similar to Barnes and Posner [9], we augment our training data with random rotations of up to 0.26 radians. Our sliding-window implementation [7] takes on average 0.07 seconds, 0.13 seconds, and 0.18 seconds for window sizes of 2, 3, and 4, respectively. Since the radar sensor spins at 4 Hz, our current implementation is real-time capable. During evaluation, we use the timestamp of each measurement to do continuous-time estimation with STEAM to compensate for motion distortion. For more information on this approach, we refer readers to the work of Sean Anderson et al. [5] [6].

B. Oxford Radar RobotCar Dataset

Here, we compare our radar odometry results with other point-based radar odometry methods including Cen and Newman [15] (Cen RO), Barnes and Posner [9] (Under the Radar), CFEAR [2], and Burnett et al. [14] (MC-RANSAC). Following these previous works, we report our results using the KITTI odometry metrics [23], which average the relative

<table>
<thead>
<tr>
<th>Methods</th>
<th>Supervision</th>
<th>Trans. Error (%)</th>
<th>Rot. Error (deg/1000m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnderTheRadar [9]</td>
<td>Supervised (L)</td>
<td>2.0593</td>
<td>6.7</td>
</tr>
<tr>
<td>MC-RANSAC [14]</td>
<td>Unsupervised (HC)</td>
<td>3.3190</td>
<td>10.93</td>
</tr>
<tr>
<td>CFEAR [2]</td>
<td>Unsupervised (HC)</td>
<td>1.76</td>
<td>5.0</td>
</tr>
<tr>
<td>HERO (Ours)</td>
<td>Unsupervised (L)</td>
<td>1.9879</td>
<td>6.524</td>
</tr>
</tbody>
</table>

### TABLE I

Radar Odometry Results. (HC): Hand-crafted, (L): Learned.

* https://github.com/utiasASRL/steam

* On an Nvidia Tesla V100 GPU and 2.2 GHz Intel Xeon CPU.
There is no image in the text.

In this section, we provide additional experimental results for our radar odometry using 100 km of driving obtained in an urban environment\(^3\). The dataset was collected using the platform shown in Figure 1 which includes a Velodyne Alpha-Prime (128-beam) lidar, Navtech CIR204-H radar, FLIR Blackfly S monocam camera, and Applanix POSLV GNSS. This Navtech sensor has a range resolution of 0.0596 m and a total range of 200 m, compared to the 0.0438 m resolution and 163 m range of the Navtech sensor used in the Oxford dataset. We do not require groundtruth position data to train our network, but obtain it for the test sequences in order to calculate KITTI odometry drift metrics. Groundtruth positioning is obtained from the post-processed GNSS results and is accurate to within 12 cm.

The 100 km of total driving data is divided into 11 individual sequences, each approximately 9 km in length. We use 7 sequences for training, 1 for validation, and 3 for testing. Of the 3 test sequences, two were taken during a snow storm, and the other was taken on a sunny day.

Figure 6 illustrates the large differences in weather conditions. Between the sunny and snowy days, there were significant visual appearance changes. Most of the lane lines are not visible during the snow storm. Even more stark is the contrast between the lidar data taken on the two days (after the ground plane is removed). During the snow storm, the lidar scan is littered with a large number of detections associated with snow flakes. Although we do not provide experimental results for this, it seems probable that the accuracy of a lidar odometry

\(^3\)We plan on making our dataset publicly available in the next year.
Fig. 6. This figure illustrates the differences in camera, lidar, and radar data between a sunny day and a snow storm. During snowfall, the lidar sensor becomes littered with noisy snowflake detections but the radar data appears unperturbed.

system would suffer under these conditions. However, Charron et al. [17] have shown that it is possible for these effects to be compensated. Unsurprisingly, it is difficult to discern the difference between the radar scans, other than the movement of large vehicles.

The robustness of radar to inclement weather is also exemplified in the odometry drift results reported in Table III. In fact, the drift rates on the snowy days were lower than on the sunny day. We postulate that the error increases on the sunny day due to the increase in the vehicle's speed. It is interesting to note that our rotation drift metrics here are lower than on the Oxford dataset. While it is difficult to point to a single factor as the reason for this difference in performance, we note that our trajectory involves less turns and we use a different source of groundtruth for testing. In Figure 7, we provide a plot of our odometry results during a snowy sequence. Here, we have plotted the results of training on the Oxford dataset and testing on this sequence, as well as training on the Boreas dataset and testing on this sequence. There is a noticeable increase in the drift rate when transferring from the Oxford dataset to this Boreas dataset. We hypothesize that this difference is mainly due to the different range resolutions of the sensors.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Weather</th>
<th>Trained On</th>
<th>Trans. Error (%)</th>
<th>Rot. Error (deg/1000m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>01-26-10-59</td>
<td>Snow</td>
<td>Boreas</td>
<td>2.003</td>
<td>5.599</td>
</tr>
<tr>
<td>01-26-11-22</td>
<td>Snow</td>
<td>Boreas</td>
<td>1.980</td>
<td>5.288</td>
</tr>
<tr>
<td>02-09-12-55</td>
<td>Sun</td>
<td>Boreas</td>
<td>2.073</td>
<td>5.886</td>
</tr>
<tr>
<td>01-26-10-59</td>
<td>Snow</td>
<td>Oxford</td>
<td>2.355</td>
<td>8.137</td>
</tr>
<tr>
<td>01-26-11-22</td>
<td>Snow</td>
<td>Oxford</td>
<td>2.112</td>
<td>6.442</td>
</tr>
<tr>
<td>02-09-12-55</td>
<td>Sun</td>
<td>Oxford</td>
<td>2.546</td>
<td>8.911</td>
</tr>
</tbody>
</table>

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we applied the ESGVI parameter learning framework to learn radar features for odometry using only the on-board radar data. Our odometry performance on the Oxford Radar RobotCar Dataset approaches the current state of the art, which is a hand-crafted method [2]. We provided additional experimental results on 100 km of driving taken in an urban setting. Within this dataset, we demonstrated the effectiveness of radar odometry during heavy snowfall.

In recent years, robotics research has become increasingly reliant on high-quality datasets for training deep learning algorithms. The size and scope of these datasets is a limiting factor in the performance of many robotic systems. However, collecting and annotating these datasets is an expensive and labour-intensive process. For this reason, it is important that research in robotics trend towards solutions that are more data efficient. By foregoing the need for groundtruth pose information, our unsupervised architecture enables large quantities of training data to be collected with relative ease.

Our framework is a hybrid of modern deep learning and classic probabilistic state estimation. Deep learning can be used to process rich sensor data while probabilistic estimation can be used to handle out-of-distribution samples through outlier rejection schemes. Furthermore, our modular framework enables many future extensions and avenues of research. For example, we plan on incorporating IMU factors into our estimator, which should be straightforward with our probabilistic estimator.
We would like to thank Applanix Corporation and the Natural Sciences and Engineering Research Council of Canada (NSERC) for supporting this work. We personally thank Andrew Lambert at Applanix and Haowei Zhang at ASRL for helping with processing and collecting data. We thank General Motors for their donation of the Buick (vehicle).

REFERENCES

APPENDIX

TABLE IV
Evaluation on 7 sequences from the Oxford Dataset. Performance is reported as translational drift (%) / rotational drift (deg/1000m) using the common KITTI odometry metric. This table is restricted to algorithms that have been tested on similar sequences. Cen RO, Kung RO, and MC-RANSAC are hand-crafted algorithms. Masking by Moving, Under the Radar, and HERO are all learning-based algorithms. HERO is currently the only unsupervised learning-based radar odometry approach. *Tested on all 32 sequences. **Uses dense correlation between scans.

<table>
<thead>
<tr>
<th>Method</th>
<th>Evaluation</th>
<th>Sequences</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10-14-02</td>
<td>11-12-26</td>
</tr>
<tr>
<td>Cen RO [15]</td>
<td>[9]</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Kung RO* [34]</td>
<td>[34]</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Masking by Moving**</td>
<td>[11]</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Under the Radar [9]</td>
<td>[9]</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>MC-RANSAC [14]</td>
<td>Ours</td>
<td>3.43/11.2</td>
<td>3.72/12.9</td>
</tr>
<tr>
<td>HERO (Ours)</td>
<td>Ours</td>
<td>1.86/5.9</td>
<td>1.85/6.2</td>
</tr>
</tbody>
</table>

Fig. 8. These results illustrate the performance of HERO during each of the test sequences. See Figure 5 for HERO’s performance on 10-14-02.

TABLE V
An evaluation on 8 sequences from the Oxford Dataset. This experiment required a new train/validation/test split. 16-13-42 was used for validation. Performance is reported as translational drift (%) / rotational drift (deg/100m) using the common KITTI odometry metric. *Uses loop closures.

<table>
<thead>
<tr>
<th>Method</th>
<th>Evaluation</th>
<th>Sequences</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hong RO [27]</td>
<td>[27]</td>
<td>2.16/0.6</td>
<td>2.32/0.7</td>
</tr>
<tr>
<td>Hong SLAM* [27]</td>
<td>[27]</td>
<td>1.96/0.7</td>
<td>1.98/0.6</td>
</tr>
<tr>
<td>CFEAR [2]</td>
<td>[2]</td>
<td>1.65/0.48</td>
<td>1.64/0.48</td>
</tr>
<tr>
<td>HERO (Ours)</td>
<td>Ours</td>
<td>2.14/0.71</td>
<td>1.77/0.62</td>
</tr>
</tbody>
</table>

An evaluation on 8 sequences from the Oxford Dataset. This experiment required a new train/validation/test split. 16-13-42 was used for validation. Performance is reported as translational drift (%) / rotational drift (deg/100m) using the common KITTI odometry metric. *Uses loop closures.