applied sciences

MDPI

Review

Survey of Deep Learning-Based Methods for FMCW Radar Odometry and Ego-Localization

Marvin Brune *, Tobias Meisen and André Pomp

Institute for Technologies and Management of Digtial Transformation, University of Wuppertal, Gaußstraße 20, 42119 Wuppertal, Germany; meisen@uni-wuppertal.de (T.M.); pomp@uni-wuppertal.de (A.P.)
* Correspondence: marvin.brune@uni-wuppertal.de

Abstract: This paper provides an in-depth review of deep learning techniques to address the challenges of odometry and global ego-localization using frequency modulated continuous wave (FMCW) radar sensors. In particular, we focus on the prediction of odometry, which involves the determination of the ego-motion of a system by external sensors, and loop closure detection, which concentrates on the determination of the ego-position typically on an existing map. We initially emphasize the significance of these tasks in the context of radar sensors and underscore the motivations behind them. The subsequent sections delve into the practical implementation of deep learning approaches, strategically designed to effectively address the aforementioned challenges. We primarily focus on spinning and automotive radar configurations within the domain of autonomous driving. Additionally, we introduce publicly available datasets that have been instrumental in addressing these challenges and analyze the importance and struggles of current methods used for radar based odometry and localization. In conclusion, this paper highlights the distinctions between the addressed tasks and other radar perception applications, while also discussing their differences from challenges posed by alternative sensor modalities. The findings contribute to the ongoing discourse on advancing radar sensor capabilities through the application of deep learning methodologies, particularly in the context of enhancing odometry and ego-localization for autonomous driving applications.

Keywords: radar; SLAM; autonomous driving; ego-localization; odometry

Citation: Brune, M.; Meisen, T.; Pomp, A. Survey of Deep Learning-Based Methods for FMCW Radar Odometry and Ego-Localization. *Appl. Sci.* **2024**, *14*, 2267. https://doi.org/10.3390/app14062267

Academic Editor: Atsushi Mase

Received: 4 January 2024
Revised: 28 February 2024
Accepted: 29 February 2024
Published: 8 March 2024

1. Introduction

Simultaneous Localization and Mapping (SLAM) stands as a foundational technology in the realms of robotics and autonomous systems. Its primary function involves enabling an autonomous system to not only construct a comprehensive map of its surroundings, but also to accurately determine its own position within this map. Central to any SLAM algorithm are two pivotal concepts: odometry, which forecasts the system's ego-motion, and ego-localization, the skill to establish its position within the map using external sensor data.

Passive and active perception-based sensors, such as cameras, lidars, sonars, and radars, are commonly applied in a SLAM approach [1]. These sensors have the advantage of deriving accuracy from observations. Maps can be constructed from these sensors to gain a spatial understanding of the environment. In an ideal world, information derived from active sensors, such as lidars, sonars, and radars, denote absolute precision for a measurement, and the odometry and ego-localization of a system can be determined solely from this information. However, due to various factors, the detections are not always consistent or accurate [2]. Passive positional sensors, such as a Global Navigation Satellite System (GNSS) or an Inertial Navigation System (INS) can provide additional information about the position and dynamics of the system, but are unable to capture the spatial environment on their own. Additionally, GNSS sensors also suffer from poor performance in common application areas, such as indoor scenarios, tunnels, and in between sky scrapers due to poor signal reception [3]. In such scenarios, perception-based ego-localization is preferred.

Positional sensors such as GNSS and INS are often included in autonomous vehicle applications besides a perception sensor, which will be discussed later.

Perception-based SLAM approaches can be categorized according to the type of sensor they rely on, with a primary distinction being between active range-based and passive visual-based methodologies. Visual SLAM (VSLAM) methods specifically employ onboard cameras to depict their surroundings. Despite the considerable research on VSLAM techniques, they encounter difficulties in adverse weather conditions like heavy rain or fog, and in varying lighting situations such as intense sunlight or during the night. These challenges extend to issues concerning accurate depth perception [4,5].

Range-based techniques utilize sensors such as lidar, sonar, and radar to operate. Sonar finds its main utility in underwater scenarios [6,7] whereas lidars and radars are predominantly employed in both indoor and outdoor applications, particularly in the context of autonomous vehicles. While lidars offer denser detection capabilities, radars bring their own set of distinctive advantages. Lidars tend to be more expensive than radars and are generally less resilient in adverse weather conditions. Although hybrid approaches that leverage both radars and lidars for SLAM tasks exist, the focus in literature and industry remains on the employment of a solitary range-based sensor [5,8]. In practice, a range-based sensor is often integrated beside visual sensors to provide depth information and assist in visually impaired scenarios. Another motivation in incorporating multiple sensors into a complete SLAM is to avoid the occlusion problem that arises when using a solitary perception sensor. When an object is obstructed by another object, or an object moves out of the frame of one sensor, another sensor might be able to spot the object. Radars are especially relevant for such a task, since radars can detect objects that appear occluded in lidars or visual data [9]. However, radar data are very susceptible to "ghost detections" [9], where a radar detects objects that are not in the scene. These detections are mostly caused by multi-path propagation [10] effects. Similar to the occlusion problem in computer vision, a deep neural network can learn a mask to filter out non-relevant objects and regions. An extra sensor, such as lidar or camera, can also provide additional spatial information to the system.

The framework for classifying SLAM approaches was introduced in 1995 through the seminal works by Durrant-Whyte et al. [11,12]. In the realm of perception-based SLAM methods, a central concept involved manually designing features to facilitate the alignment of keypoints within image or pointcloud representations. However, much like the progress witnessed in various other fields, such as natural language processing [13] and computer vision [14], deep learning methods have been developed to address the challenges faced in a SLAM method.

In the field of computer vision, deep learning has been applied thoroughly to address a variety of issues for the SLAM problem, including keypoint detection, ego-motion estimation, structure-from-motion, and ego-localization [15–18]. Analog methods have been developed for lidar data that work on a point cloud representation rather than a grid representation [19–21]. The type of deep learning architecture consists of a range of modules from standard spatial and temporal layers such as convolutional neural networks (CNNs) [22] and recurrent neural network (RNN) such as long short-term Memory layers (LSTMs) [23] and Gated recurrent units (GRUs) [24], to sophisticated graph neural-networks (GNNs) [25] and transformer architectures [26]. Other optimization methods, such as formulating an appropriate loss, multi-task learning, and optimization learning techniques have been thoroughly investigated for cameras and lidar. The exact architecture and training routine depends on the available data, and on the sensors present in the system [27–29].

Given the increasing prominence of radar as a robust, cost-effective, and dependable perception sensor [5,9,30], there is a distinct need to undertake a comprehensive review of deep learning approaches specifically tailored toward radar-based SLAM methodologies.

Our main contributions are as follows:

1. We provide an introduction to the formulation of the SLAM problem and radar processing.
2. We delve into a detailed examination of the latest deep learning techniques employed in radar-based odometry and localization.
3. We explore the current methods to integrate radar data into deep learning modules to enhance visual/lidar odometry and ego-localization accuracy.

Figure 1 provides an overview of the survey paper proposed in this study. Section 2 introduces the overarching problem of sensor-based odometry and ego-localization. Moving forward, Section 3 references existing survey papers, setting the context for our work. In Section 4, our attention is directed towards fundamental radar processing, exploring diverse setups and methods of data representation for radar data. Section 5 consolidates the latest trends by examining the application of deep learning methods in radar-based odometry and ego-localization pipelines. Notably, we extend our discussion beyond solely radar data to include the integration of fusion and cascading network structures to account for the necessity for a complete and reliable SLAM approach. We also compare the evaluation methods of the models. To conclude, Section 6 offers a comprehensive summary and discussion of potential future trends in the field.

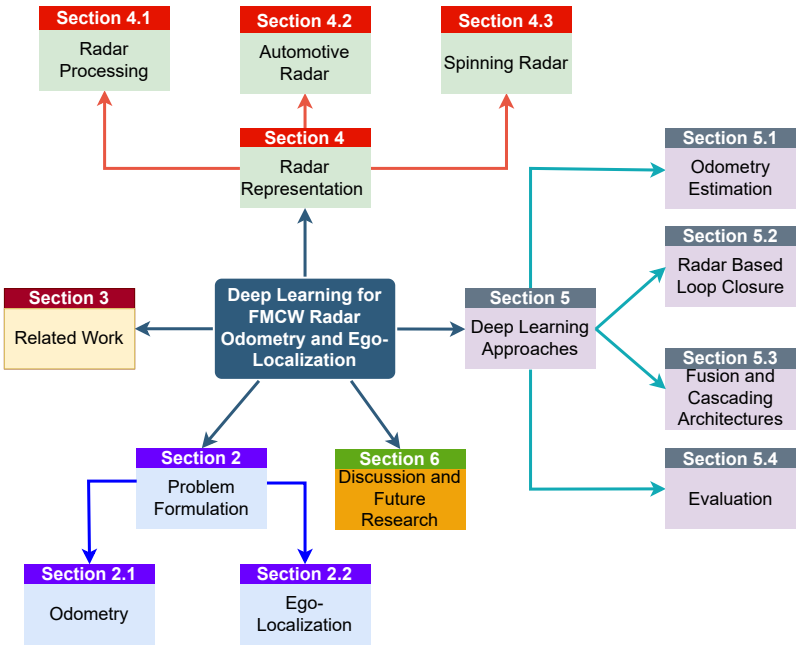

Figure 1. Overview of the proposed survey paper.

2. Problem Formulation

In this section, we present fundamental concepts and the sequential stages involved in odometry estimation and ego-localization within radar-based SLAM. Additionally, we highlight key methods employed in classical radar SLAM.

SLAM constitutes an estimation challenge wherein the objective is to refine the collection of all ego poses $\boldsymbol{X}_{1:t} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_t\}$ at times 1 through t and n landmark positions $\boldsymbol{M} = \boldsymbol{m}_{1:n} = \{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_n\}$ recorded in a global map $\boldsymbol{M}$. This refinement is accomplished by utilizing observed data encompassing perceptions $\boldsymbol{Z}_{1:t} = \{\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_t\}$, gathered odometry readings $\boldsymbol{U}_{1:t} = \{\boldsymbol{U}_1, \ldots, \boldsymbol{U}_t\}$, and an initial starting point $\boldsymbol{X}_0$. Mathematically, the probabilistic formulation is articulated as defined by Grisetti et al. [31] as:

$$p(\boldsymbol{X}_{1:t}, \boldsymbol{M} | \boldsymbol{Z}_{1:t}, \boldsymbol{U}_{1:t}, \boldsymbol{X}_0) \tag{1}$$

For convenience, the starting position $\boldsymbol{X}_0$ from Equation (1) will be omitted from now on. The ego-position and odometry measurements are commonly depicted using 2D

or 3D special Euclidean group transformation matrices $\boldsymbol{T} \in SE(2)$ or $\boldsymbol{T} \in SE(3)$. Such transformation matrices consist of a translation vector t and a rotation matrix R which can be represented with homogeneous coordinates in Equation (2) as

$$\boldsymbol{T} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{2}$$

The fundamental objective of any SLAM methodology is to determine the Maximum A Posteriori (MAP) state for trajectories $\boldsymbol{X}_{1:t}$ and the landmark map $\boldsymbol{M}$. This objective can be formally expressed as:

$$\{\boldsymbol{X}^*, \boldsymbol{M}^*\} = \underset{\{\boldsymbol{X},\boldsymbol{M}\}}{\operatorname{argmax}}\, p(\boldsymbol{X}_{1:t}, \boldsymbol{M} | \boldsymbol{Z}_{1:t}, \boldsymbol{U}_{1:t}) \tag{3}$$

A commonly employed approach to solving Equation (3) involves making an assumption of the static global map and leveraging the Markov property. This allows the SLAM problem to be conceptualized as a dynamic Bayesian network (DBN) [31]. Within this framework, the initial pose $\boldsymbol{X}_0$, observations $\boldsymbol{Z}_{1:t}$ and odometry estimations $\boldsymbol{U}_{1:t}$ are considered as observable variables. Meanwhile, the latent variables of the model encompass $\boldsymbol{X}_{1:t}$ (sequential poses) and the map $\boldsymbol{M}$. The structure of the DBN is characterized by two main components: a state transition model denoted as $p(\boldsymbol{X}_t | \boldsymbol{X}_{t-1}, \boldsymbol{U}_t)$ which signifies that a new pose is contingent on the previous pose and the current odometry measurement. The second component is an observation model denoted as $p(\boldsymbol{Z}_t | \boldsymbol{X}_t, \boldsymbol{M})$, indicating that observations are dependent on the current pose and the overall map configuration.

A DBN can also be portrayed using a graphical representation, such as that presented in [32], where nodes correspond to poses $\boldsymbol{X}_t$. Within this graph, edges serve as spatial constraints connecting poses. These constraints can be established based on odometry measurements $\boldsymbol{U}_t$ between poses or through the alignment of two distinct observations $\boldsymbol{Z}_t$ taken at different poses. In practical implementations, the observation edges play a crucial role in the graphical representation by functioning as regulators. Their purpose is to rectify any potential accumulation of errors or drift that might arise from the utilization of odometry edges alone.

Graph optimization is a prevailing technique employed to determine the optimal global solution, considering multiple outcomes from scan matching. Depending on the specific characteristics of a SLAM algorithm, the edges accumulated through odometry and observations might not align seamlessly. This discrepancy can arise due to assorted factors, encompassing both hardware and software intricacies. Despite these potential inconsistencies, the ultimate objective remains the establishment of a globally coherent trajectory that accounts for all observations and information available. A typical SLAM procedure is shown in Figure 2.

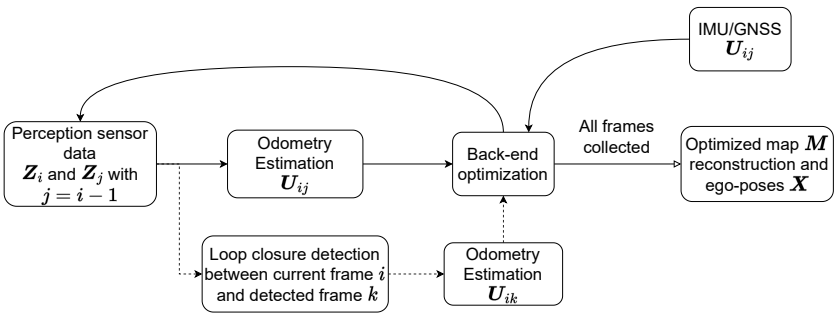

Figure 2. Typical setup of a SLAM approach. Sensor data are used to estimate the transformation matrix between frames. If a loop closure is detected, a transformation matrix between the detected frame and current frame is added to the back-end optimization scheme. Finally, an optimized map and ego-poses are generated after the back-end optimization process. IMU/GNSS data can be inserted as additional edges in the back-end optimization.

2.1. Odometry

Various sensors can serve to establish an odometry edge within the graph representation. An inertial measurement unit (IMU) achieves this by utilizing internal sensors like gyroscopes, accelerometers, and magnetometers. Similarly, GNSS sensors like a GPS, differential GPS (DGPS) or INS with integrated GPS can also provide odometry measurements from the pose difference computed at two different locations. However, passive IMU and GNSS sensors lack environmental context and are thus vulnerable to external factors such as signal interference or incorrect readings, for example due to a stuck vehicle, despite wheel movement detected by the IMU.

On the other hand, scan-based odometry methods deduce the vehicle's ego-motion by analyzing environmental data from perception sensors like radar, lidar, and/or vision. Unlike IMU-predicted odometry, sensor-based odometry aligns consecutive scans, leading to a consistent odometry estimate that is grounded in the external environment. However, relying on external sensors introduces its own set of challenges, which can potentially yield inaccurate estimations depending on the specific sensor. In practice, the IMU/GNSS data and scan-matching results are often fused in a Kalman-Filter approach [9]. Here, it needs to be mentioned that any IMU needs to be re-calibrated from time to time due to various factors, including mechanical deterioration, change in environment, or temperature changes [33]. As such, odometry estimations from other sensors are crucial in a SLAM method, allowing accurate prediction of the ego-motion.

In the realm of radar-based scan matching, a prevalent technique involves range-based feature matching [9]. Radar data are typically represented in the form of a point cloud or a 2D bird's-eye view (BEV) image. The goal is to align consecutive scans to identify the most probable transformation. While deep learning methods can be used to determine the ego-transformation between successive radar scans, traditional techniques continue to find more widespread application. These conventional methods entail the use of manually designed features. Examples include Oriented FAST and rotated BRIEF (ORB)-features [34] and scale-invariant feature transform (SIFT)-features [35] for visual data, as well as Conservative Filtering for Efficient and Accurate Radar Odometry (CFEAR) [36] features tailored for radar data. These features are expressed as point detections. The process of scan matching involves identifying these features in single scans and matching them across consecutive scans. This is accomplished through an iterative closest point (ICP)-based approach [37,38], which is often integrated with a random sample consensus (RANSAC) [39] technique to discard outliers.

For laser scan observations, alternate methods like the normal distribution transform (NDT) [40] are commonly employed to calculate the ego-transformation matrix between successive scans. For accurate alignment of sensor measurements, it is essential to exclusively include relevant stationary features. Introducing non-stationary features into the ICP process can distort the vehicle's ego-motion estimation, yielding inaccurate outcomes [41]. Consequently, it is crucial to filter out detections originating from irrelevant features. While this issue has been extensively addressed for cameras and lidar sensors, radar sensors largely evade this concern due to the captured Doppler feature. Although the Doppler component solely provides the radial velocity of a feature, it can be employed to deduce the true velocity of a detection. This allows the removal of non-stationary features before applying an ICP-based approach, ensuring more accurate results [41,42].

2.2. Ego-Localization

Odometry estimation plays a crucial role in accurately mapping consecutive scans of the environment. However, regardless of the precision of the odometry measurement and the sensors used, there is an inevitable build-up of global drift over time. This drift can lead to distortions in the generated map and, thus, inaccuracies in global pose estimations [9]. To mitigate such drift issues, ego-localization is employed in the form of loop closure detections to identify previously visited locations. When an already visited location is detected, a scan matching procedure is carried out between the present and past scan or

map, resulting in an adjustment of the current pose based on the matching outcome. In graph optimization approaches, loop closure transformations are included as additional edges in the graph structure.

Various techniques are employed in the detection of potential loop closures. Some methods employ the usage of positional sensors such as GPS, while others match a new scan to a scan from a set of stored scans to determine if an edge is added to the graph optimization. One common approach utilizes the Bag-of-Words (BoW) method [43]. This technique proves valuable for efficiently comparing perception scans, a critical aspect in the detection of loop closures within SLAM and analogous systems. The BoW technique treats each scan as a collection of visual words, succinctly represented as a single vector. These visual words encompass features such as edges, corners, and texture sections, extracted through algorithms like SIFT, SURF, ORB, or a trained network.

Initially, a dictionary is crafted by aggregating features from diverse scans and organizing them through a cluster analysis, such as k-means clustering. Each cluster center subsequently serves as a distinct 'visual word.' When a new scan is recorded, its features are extracted and matched against the visual words, or cluster centers, present in the dictionary. This results in the representation of the scan as a vector, illustrating the frequency of each visual word within the scan. During the ego-localization process, a BoW is employed to identify similarities between the current scan and known scenes, enabling the estimation of the position concerning previously visited locations. If the similarity between the current vector and a vector from the dictionary succeeds a previous defined threshold, scan matching is performed between the two scans to determine the transformation matrix and an edge is added to the graph. However, these edges have to be carefully chosen, as incorrect edges can lead to large deteriorations of the generated map [11,12].

3. Related Work

A plethora of survey papers offer in-depth insights into a diverse array of scan matching and odometry estimation methods, addressing different sensor and method types, as illustrated in Table 1. These survey papers were found by combining the terms "Survey" or "Review' with a task or sensor-specific term such as: "SLAM", "VSLAM", "lidar", "Survey", "Ego-localization", "Localization", "Deep learning", "Radar", "Odometery" using Google Scholar and Web of Science. Additional papers were found with a backward search on the found surveys.

While there are numerous survey papers dedicated to odometry and localization in SLAM, none specifically delve into the application of deep learning to the radar modality. A recent contribution by Louback da Silva Lubanco et al. [44] lays the groundwork for a foundational understanding of classic radar-based odometry.

Geng et al. [45] discuss advancements in deep learning applied to radar perception, although their focus does not specifically center on odometry or place recognition. Conversely, the review conducted by Yao et al. [4] revolves around radar-vision fusion specifically for object detection and segmentation.

The survey papers authored by Zhou et al. [9] and Harlow et al. [5] provide thorough insights into radar pre-processing schemes and deliver a detailed overview of state-of-the-art methods for radar perception tasks within the domain of autonomous driving. Notably, these reviews predominantly concentrate on radar perception concerning object detection and segmentation. They touch upon only a limited number of deep learning-based methods for odometry estimation and localization.

Table 1. Overview of relevant survey papers. Primary Sensor: R, L, and C stand for Radar, lidar, and Camera sensors. ✓ means the survey included the SLAM component, (✓) means it was partially included and × that it was not included.

Paper	Year	Primary Sensor	Deep Learning Focus	Odometry	Localization
[44]	2022	R	×	✓	×
[46]	2018	R,L,C,GPS	×	×	✓
[47]	2021	L,C	✓	×	✓
[48]	2021	L,C	✓	×	✓
[49]	2022	R,L,C	✓	×	✓
[29]	2023	L,C	✓	✓	✓
[45]	2021	R	✓	×	×
[4]	2023	R,C	✓	×	×
[50]	2022	L,C	×	✓	✓
[51]	2022	L,C	✓	✓	✓
[52]	2021	L	✓	✓	✓
[53]	2021	L	×	✓	✓
[54]	2022	C	✓	✓	✓
[55]	2015	C	×	×	✓
[56]	2020	C	✓	✓	×
[57]	2019	C	✓	✓	✓
[58]	2023	L,C	(✓)	✓	✓
[8]	2020	L,C	✓	✓	✓
[27]	2023	C	✓	✓	✓
[59]	2023	C	✓	✓	✓
[28]	2023	C	✓	✓	✓
[9]	2022	R	(✓)	✓	✓
[5]	2023	R	(✓)	✓	✓
Ours	2023	R	✓	✓	✓

The work by Saleem et al. [29] closely aligns with our survey paper, as these authors delve into the fundamentals of SLAM issues and present numerous deep learning techniques applied to lidar and visual sensors. It is noteworthy, however, that their coverage of radar is confined to potential fusion scenarios with lidar or visual sensors.

Besides these survey papers focusing on radar, many others focus on either visual or lidar-based SLAM methodology. Kuutti et al. [46] concentrate on non-learning based localization methods for various perception sensors. Extending this work, Arshad et al. [47] investigate how deep learning techniques facilitate loop closure identification in lidar and VSLAM. Shifting focus, Roy et al. [48] specialize in deep learning techniques tailored for indoor localization and embedded systems. Additionally, Yin et al. [49] provide an extensive coverage of place recognition and loop closure detection across sensor modalities and application domains.

In the realm of sensor fusion, Xu et al. [50] highlight the integration of lidar-based sensors with other modalities, primarily leveraging visual features. On a similar note, Chghaf et al. [51] concentrate on lidar and VSLAM, briefly addressing the fusion of radar with other modalities.

Huang et al. [52] and Khan et al. [53] predominantly center their reviews on lidar-SLAM. In the context of VSLAM, several comprehensive review papers, including [54–57,60], provide detailed overviews, but lack any mention of radar sensors.

The exploration of deep learning across diverse sensor modalities is addressed in the works by Placed et al. [58] and Chen et al. [8], with a specific focus on lidar and VSLAM. They particularly emphasize belief-space planning and deep reinforcement learning. In a related context, Zhang et al. [27] and Favorskaya et al. [59] delve into the role of deep learning in VSLAM approaches, investigating its applications in fundamental SLAM tasks such as pose optimization and mapping.

Mokssit et al. [28] concentrate on specific stages of VSLAM methods, encompassing depth estimation, optical flow, odometry, loop closure, and end-to-end learning. Additionally, the review by Zeng et al. [61] explores the application of the transformer architecture in point cloud representations within the context of VSLAM.

4. Radar Representation

In this section, we focus on radar processing and the diverse ways in which radar data can be represented. As we dive into standard signal processing for radars, we focus on the two most commonly applied radar configurations for autonomous vehicles; automotive and spinning radar.

4.1. Radar Processing

Radar sensors have found utility across various applications since the 1930s and have gained particular popularity for perception tasks. This preference stems from their robustness and relatively affordable cost [5,9].

A radar sensor functions by emitting and receiving frequency-modulated continuous waves (FMCW), usually within frequency bands centered at 24 GHz or 77 GHz with a linearly increasing frequency. To achieve this, a frequency mixer is employed, combining the transmitted and received signals to manage high-speed sampling requirements. Subsequently, a low-pass filter is applied to eliminate the summed components, resulting in the extraction of an intermediate-frequency signal (IF). This IF signal can be captured by an Analog-to-Digital Converter (ADC), yielding a discrete-time complex ADC signal. The process of determining an object's range involves analyzing the beat frequency, which emerges due to the time difference between transmission and reception [9].

Simultaneously, the Doppler velocity is ascertained by examining the phase shift within the IF signal. This phase shift arises due to the consecutive generation of FMCW waves, also referred to as chirps. The dimension of the sampling points within a single chirp is referred to as the *fast time*, and the dimension of a single chirp index across multiple chirps in a single frame is referred to as *slow time*. Under the assumptions that the range variations in slow time caused by target motion can be neglected due to a short frame time and by utilizing large frequency modulation to neglect the Doppler frequency in fast time, the range and Doppler calculations can be decoupled. A pivotal outcome of this process is a complex-valued data matrix, achieved through two discrete Fourier transforms carried out over the fast-time and slow-time dimensions. Known as the range–Doppler (RD) map, this matrix depicts power readings corresponding to specific range and Doppler values. This map offers valuable insight into the presence and motion of objects detected within the radar's scope [5].

Every FMCW radar employs the outlined scheme to calculate range and Doppler components. However, techniques diverge when it comes to determining the object's direction of arrival (DOA). This distinction hinges on the radar's positioning and configuration, leading to two main categories of methods: *automotive radars* (AR) and *spinning radars* (SR). Figure 3 illustrates how each radar sensor configuration can be placed on a vehicle.

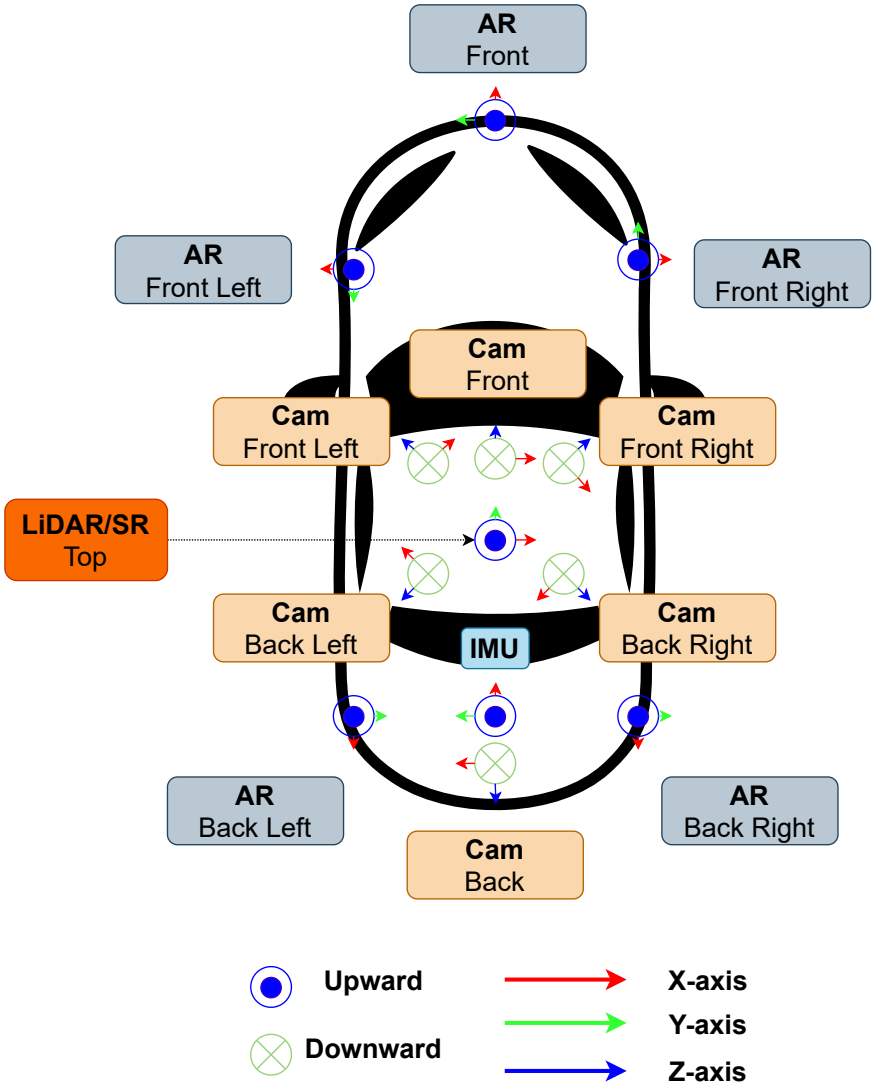

Figure 3. AR stands for automotive radar sensors and SR stands for spinning a radar sensor. A single spinning radar sensor is installed on top of the vehicle, while multiple radar sensors are installed in front and in the corners or sides of the vehicle in an automotive radar configuration.

Spinning radar configurations have the advantage that they only consist of a single sensor placed on top of the vehicle. A sole external calibration matrix is used to project the objects into the vehicle's frame of reference. For an automotive radar configuration, radar sensors, typically 4–8 small chips, are placed at different positions around the vehicle. Here, the information recorded by the sensors needs to be transformed into the same coordinate system, often into the vehicle coordinate system, to match the same detections recorded by different sensors. As such, point cloud representations are favored when working with AR configurations [5,9].

Table 2 provides an overview of presently utilized open-source radar datasets that are explicitly employed for ego-motion or ego-localization purposes using radar sensors. The datasets, discovered through the GitHub repository [62], were supplemented by additional datasets identified in Section 5.

Table 2. Radar datasets applied for deep learning SLAM methods. Type: SR and AR stand for spinning radar and automotive radar. Scenarios: U,S,H,P,I,T,W stand for urban, suburban, highway, parking lot, indoors, tunnel, and waterways. Other sensors: C,L,O,S stand for camera, LiDAR, internal odometry sensor, and synthetic aperture radar (SAR). ✓ datasets include a Doppler measurement, and × do not.

Name	Type	Data Representation	Doppler	Scenarios	Other Sensors
Oxford [63]	SR	RA	×	U	CLO
Boreas [64]	SR	RA	×	S	CLO
MulRan [65]	SR	RA	×	US	LO
RADIATE [66]	SR	RA	×	USHP	LO
Nuscenes [67]	AR	PC	✓	USH	CLO
View-of-Delft [68]	AR	4D PC	✓	USH	CLO
ColoRadar [69]	AR	ADC,PC	✓	SIT	CLO
USVInland [70]	AR	PC	✓	W	CLO
Zendar [71]	AR	ADC, RD, PC	✓	U	CLOS
EU Long-term [72]	AR	PC	✓	U	CLO
Astyx [73]	AR	3D PC	✓	SH	CLO
K-Radar [74]	AR	RAD, 4D PC	✓	USHP	CLO
RadIAL [75]	AR	ADC	✓	USH	CLO

While mathematical expressions and derivations for variables like the IF signal, range, Doppler velocity, and DOA exist, these details are not covered here. This omission is due to the fact that radar-based odometry and ego-localization methods are typically applied post Constant False Alarm Rate (CFAR) processing, and have been covered extensively in other works [5,9]. Additionally, raw data representations are infrequently included in current publicly investigated datasets, making models dependent on the pre-processing of each respected dataset.

Here, it also has to be noted that pre-processing of radar data is an active ongoing research field. One further issue with radar sensors is the problem of mutual inference, especially when considering AR configurations. As the number of radar sensors increases, the radars can interfere with one another, potentially creating ghost object detections and a reduced signal-to-noise ratio [76]. This can be caused by radars from the same vehicle, or even from a different vehicle with a radar sensor. Methods exist that reduce the amount of interference between sensors [77], but are rarely discussed in public datasets.

It is important to highlight the differences between the mentioned setups of radar configuration in autonomous vehicles as many of the found methods in Section 5 are only applied for one or the other radar setup. These differences will be discussed next.

4.2. Automotive Radar

Automotive radars adopt a strategy of using multiple antennas, either in a single transmitting and multiple receiving (SIMO) configuration or with multiple transmitting and receiving antennas (MIMO) configuration, to determine the DOA. This determination relies on evaluating the phase change between the receiving antennas, coupled with the physical spatial separation between them. The DOA is deduced by performing an angle Fast Fourier Transform (FFT) over the receiver dimension, yielding a 3D tensor named the range–azimuth–Doppler (RAD) map. This tensor can be concatenated with the previously discussed range–Doppler (RD) map, resulting in a comprehensive representation of the direction and Doppler component of each reflection.

Traditionally, only the azimuth angle has been resolved. However, due to advancements in engineering, modern radar sensors are capable of resolving both azimuth and

elevation angles [78,79]. This enhanced capability gives rise to a 4D radar map, encompassing range, azimuth, elevation, and Doppler features.

By employing a CFAR detector on the RAD tensor, radar detections can be transformed into point clouds. However, these point clouds often contend with issues arising from road clutter, interference, and multi-path effects due to the noisy IF signal. To refine detections, additional spatial–temporal filtering methods like density-based spatial clustering of applications with noise (DBSCAN) [80] or Kalman filters can be employed.

Besides the Doppler velocity, another important feature of a radar detection is the Radar Cross Section (RCS), which depends on the properties of the reflected target. It is the intensity recorded by a receiving antenna that was deflected by an object. The intensity is dependent on the material, size, and orientation of the object. The RCS is used to differentiate between different objects. For example, cars and thick walls have a high RCS, whereas pedestrians and small objects have a general low RCS [9].

Currently, most radar-based SLAM methods lack elevation resolution. Hence, point clouds are commonly represented as 2D BEV images. To enhance object visibility and detection, multiple SIMO/MIMO sensors are often placed on a vehicle to achieve a 360-degree field of vision, often positioned at the corners of the vehicle. However, combining the features of these multiple sensors necessitates further processing due to the spatial and temporal disparities, ultimately resulting in a unified representation. Each detection of the point cloud usually consists of spatial information, Doppler and an RCS component.

4.3. Spinning Radar

In contrast to automotive radars, spinning radars utilize a single transmitting and receiving antenna. Similar to LiDAR sensors, they are positioned atop a vehicle and rotate around their vertical axis. Like any FMCW radar, these radars leverage the frequency shift between transmitted and received waves to calculate the object's range. The power of the returned signal provides valuable information regarding the object's reflectivity, size, shape, and orientation concerning the receiver. However, the DOA is determined by the current angle of the spinning radar, generating a polar power spectrum [81,82].

This 2D polar spectrum is able to generate a less sparse radar representation compared to the heavily processed point cloud generated by the automotive radar. However, due to this dense representation, the Doppler and RCS component are often not included in spinning radar open datasets. Another issue is that a spinning radar setup is unable to determine any height information due to the lack of antennas. Thus, spinning radars are generally used in 2D scenarios [81,82].

It is important to note that radar datasets differ not only in their respective type, i.e. spinning and automotive, but also in their exact sensor configurations. E.g., although two systems use an automotive radar configuration, they can drastically differ in their working range and resolution, generally depending on the price of the sensor. The same goes for spinning radars. Radars with a higher spinning frequency will provide a higher resolution range-angle image, yielding in higher quality data.

5. Deep Learning Approaches

Within this section, we discuss and compare various deep learning architectures that have been applied to radar-based odometry and localization. Subsequently, we delve into the manner in which radar data are seamlessly integrated with other sensor inputs within the context of deep learning approaches.

The investigation of various approaches involved a thorough search on Google Scholar and Web of Science. Initially, methods related to deep learning for radar-based odometry estimation or place recognition were identified on Google Scholar using keyword searches. Additional relevant methods were then found through a backwards search of the initially identified references. To complement the Google Scholar search, a similar search was conducted on Web of Science, using the same queries as those used in the Google Scholar search. The search utilized different permutations and choices of keywords

and phrases such as "Deep learning", "Learning-based", "(Un)-supervised", "Radar", "Ego-localization", "Localization", "Scan matching", "Scene flow", "Odometry", "Place recognition", "Point-based", "Sensor Fusion", and "SLAM" which were connected with the conjunctions "and" and "for". Given the relatively recent introduction of deep learning in radar-based SLAM, the search specifically considered methods published from 2016 onwards. This cutoff criterion was motivated by the famous paper by Wang et al. [83] using deep learning for visual odometry published in 2017. Figure 4 shows the number of papers investigated for odometry estimation and localization using deep learning and radar sensors. Here, the methods are also differentiated by radar setup, and if other modalities are also used by the method.

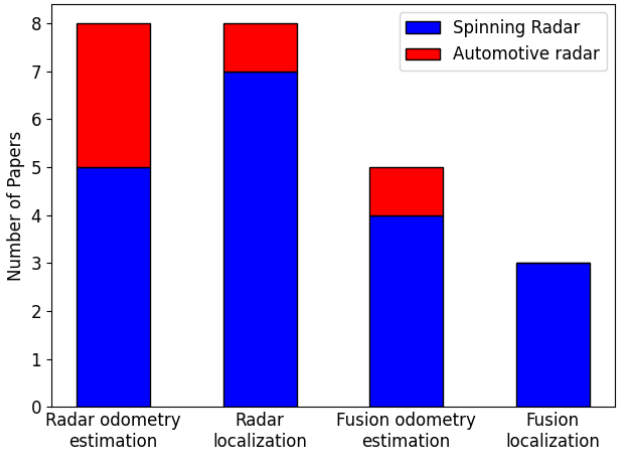

Figure 4. Number of papers utilizing deep learning for radar-based odometry estimation and localization, as well as fusion approaches with a radar sensor.

As radars are very price-effective, they can be integrated in various systems as perception sensors. These systems can vary greatly in terms of available computational hardware and use-case scenarios. For example, autonomous cars have more space to carry a large Graphics Processing Unit (GPU) for processing, whilst a cleaning robot has very limited memory. On the other hand, the car operates in a more challenging environment, operating at potentially much larger scales and ego-velocities than the cleaning robot causing greater motion distortion. The methods introduced are thus not all comparable to one another and need to be interpreted for their respected datasets and use-cases.

An overview of the investigated methods, as well as the radar representation, datasets, its task, and the usage of deep learning are shown in Table 3.

Table 3. List of deep learning methods applied for radar-based odometry and localization. SR stands for a spinning radar and AR for an automotive radar setup.

Method	Data Representation	Datasets	Task	Model Architecture and Usage
[84]	SR cartesian scan	Oxford	Odometry	CNN for masking irrelevant detections
[85]	SR cartesian scan	Oxford	Odometry	CNN for masking irrelevant detections
Fast-MbyM [86]	SR cartesian scan	Oxford	Odometry	CNN for masking irrelevant detections
UnderTheRadar [87]	SR cartesian scan	Oxford	Odometry and localization	Extraction of keypoint location, weight, and descriptor using a U-Net with multiple decoders.
HERO [88]	SR cartesian scan	Oxford, Boreas	Odometry	Extraction of keypoint location and weights using a U-Net with multiple decoders.
RaFlow [89]	AR 4D point cloud	In-house	Odometry, scene flow estimation	Multiple modules including multi-scale encoder, cost-volume layer, and flow decoder to predict scene flow.

Table 3. *Cont.*

Method	Data Representation	Datasets	Task	Model Architecture and Usage
4DRO [90]	AR 4D point cloud	View-of-Delf,In-house	Odometry	Three modules consisting of feature encoding, initial pose estimation and pose warping to predict the ego-motion.
Milli-RIO [91]	AR 3D point cloud	In-house	Odometry	LSTM layer as motion model.
Autoplace [92]	AR 3D point cloud	Nuscenes	Localization	CNN as spatial encoder and LSTM as temporal encoder. NetVLAD for local descriptor.
Kidnapped radar [93]	SR polar scan	Oxford	Localization	VGG-16 encoder and NetVLAD descriptor.
Look around you [94]	SR polar scans	Oxford	Localization	VGG-16 encoder and NetVLAD descriptor for sequential data.
[95]	SR cartesian scans	Oxford	Localization	VGG-19 or ResNet-152 as feature encoder and MLP as descriptor generator.
Radar LCD [96]	SR cartesian scan	Oxford, MulRan	Localization	U-Net feature map and keypoint generation. PointNetVLAD for descriptors.
Off the Radar [97]	SR cartesian scan	Oxford, MulRan	Localization	VAE for disentanglement of feature representation into uncertainty sources and invariant descriptor.
RadarLoc [98]	SR cartesian scan	Oxford	Global pose prediction	Self-attention and DenseNet module for feature extraction. MLP for pose estimation.
RSL-Net [99]	SR cartesian scan and satellite imagery	Oxford	Odometry	cGAN for satellite to radar. Multiple CNNs for disentanglement of rot. and trans. component.
GRAMME [100]	SR cartesian scans and camera sequences	Oxford, RADIATE	Odometry	Four modules, including an attention module for fusion, predict ego-motion for each sensor and ego-motion.
L2R [101]	SR cartesian scan	Oxford	Localization on lidar map	Lidar-to-Radar GAN to perform classic ICP and particle filter for localization.
RaLL [102]	SR cartesian scan and lidar map	Oxford, MulRan	Odometry	U-Net modules to calculate difference tensors between lidar embedding and radar scan. Differential Kalman-Filter.
Pointing the Way [103]	SR cartesian scan, radar detections, (lidar) map	Boreas	Odometry	U-Net to generate mask for detection map. Differential ICP against map data.
[104]	SR polar scan and lidar map	Oxford, MulRan	Localization	Shared U-Net encoder between radar, lidar map, and positive / negative samples. FFT as final descriptor generation.
RaLF [105]	SR cartesian scan and lidar submap	Oxford, MulRan, Boreas	Localization	RAFT feature encoder and flow field prediction. CNN for descriptor generation.
Hidden-gems [106]	AR 4D point cloud, lidar scan, camera input	View-of- Delf	Odometry and scene perception	Set conv point encoder, cost volume layer for scene flow predictions, ego-motion head, GRU for temporal updates.
Seraloc [107]	SR cartesian map	In-house	Odometry	Semantic segmentation of points.

5.1. Odometry Estimation

In the realm of deep learning methods for odometry estimation, particularly from point cloud and image representations, a central challenge revolves around identifying pertinent features and effectively tracking them across successive frames. Should these relevant features be successfully identified and consistently tracked, a differentiable point matching mechanism like the Kabsch algorithm [108] can be employed to determine the transformation matrix $T \in SE(3)$ between the frames. Yet, the crux of the matter remains in the ability to pinpoint these features and establish accurate point associations, constituting the core challenge in odometry estimation derived from sensor data.

The capacity of models to extract keypoints from both point clouds and images has undergone thorough investigation, particularly in the context of VSLAM and lidar-based SLAM methods [8,28,29]. Notably, when compared to other sensors, radar encounters a distinct challenge due to the prevalence of sparse and unreliable data.

A common method for determining transformation parameters involves a scan-matching approach, typically assuming that $\boldsymbol{T} \in SE(2)$. The optimal ego-motion parameters are estimated by learning a mask that maximizes the overlap of the noise-free representation, and trained with a loss of the form [8]:

$$\mathcal{L} = ||\hat{t} - t||_2 - \alpha||\hat{R}R^T - \mathbf{I}||_2 \tag{4}$$

where $\hat{t}$ and $\hat{R}$ are the predicted translation and rotation matrices in Equation (4). α is a fixed or learnable parameter to adjust the error between translation and rotation weight. $\mathbf{I}$ is the identity matrix.

Barnes et al. [109] utilized a masking network in a U-Net encoder–decoder architecture to effectively filter out irrelevant regions within depth images, specifically for visual odometry. Drawing inspiration from this, Aldera et al. [84] employed a similar U-Net architecture to mask specific areas within a Cartesian radar representation image. The learned mask efficiently excludes non-relevant elements, such as noise, spurious detections, and moving objects. It is important to note, however, that this process disregards the Doppler velocity of detections. In this approach, the U-Net architecture serves as the sole component responsible for distinguishing relevant landmarks.

Barnes et al. [85] also utilized a convolutional neural network (CNN) to predict a mask for spinning radar scans. This model learns to filter out all non-essential detections from the radar scan. To determine the relative pose between two scans specifically, a mask is learned for a reference scan denoted as S1 and for another subsequent scan S2. Each mask is then multiplied with its respective scan. Scan S2 is then adjusted using possible pose offsets $\Delta t = [\Delta x, \Delta y, \Delta z]$. Finally, Barnes et al. employed a brute force correlative matching technique to identify the most suitable transformation. The process involved calculating the discrete cross-correlation between the FFT of S1 and the FFT of S2, which was adjusted using the aforementioned mask and pose offsets.

Extending the foundation laid by [85], the study conducted by Weston et al. [86] takes a step forward by disentangling the search process for the rotation and translation components. This strategic decoupling serves to alleviate the computational complexity inherent in the correlative scan matching technique. Initially, the optimal angle is identified by assessing the cross-correlation of the masked scans in polar coordinates. This angle then serves as a pivotal factor in the subsequent correlative scan matching approach. By segregating the search for the rotation angle from the search for the translation component, the computational load is significantly reduced. This improvement in computational efficiency is achieved with only a minimal decline in performance when compared to the method proposed by [85].

An alternative approach to generating ego-motion predictions involves directly extracting keypoints from the data. Barnes et al. [87] implemented a U-Net encoder–decoder architecture with two decoder branches to gain insights into keypoints, weights, and descriptors. These components collectively contribute to determining a transformation matrix between two frames. The architectural design is adept at generating keypoints along with corresponding weights and descriptors for each of these keypoints. Subsequently, a matching process is applied to identify the keypoints that exhibit the most pronounced similarity. From these matched keypoints, a transformation is deduced using the Singular Value Decomposition (SVD) technique. A noteworthy aspect of their approach is the embedded differentiability within the keypoint matching and pose estimation processes. This design choice ensures that the parameters of the model remain differentiable, facilitating seamless integration with optimization processes.

Much like the approach taken in [87], Burnett et al. [88] also harnessed the capabilities of a U-Net architecture. This architecture was utilized to extract keypoints, weights, and descriptors from a spinning radar scan, ultimately facilitating the prediction of odometry across successive scans via an unsupervised methodology. The authors integrated a General Expectation-Maximization (GEM) scheme into their approach, with the primary aim of maximizing the evidence lower bound (ELBO). This combined strategy enables them to enhance the efficiency and effectiveness of their unsupervised odometry prediction process.

$$\mathcal{L} = \underbrace{\int_{-\infty}^{\infty} q(\mathbf{x}) \ln\left(\frac{p(\mathbf{x}|\mathbf{z},\theta)}{q(\mathbf{x})}\right) d\mathbf{x}}_{\leq 0} \underbrace{- \int_{-\infty}^{\infty} q(\mathbf{x}) \ln\left(\frac{p(\mathbf{x},\mathbf{z}|\theta)}{q(\mathbf{x})}\right) d\mathbf{x}}_{\text{upper bound}} \tag{5}$$

When finding update rules for the parameters, only the upper bound from Equation (5) is optimized. When assuming a multivariate Gaussian probability distribution for the posterior approximation ($q(\mathbf{x}) = \mathcal{N}(\mu, \Sigma)$), it results in a new loss.

$$V(q|\mathbf{\Theta}) = \mathbb{E}_q[\phi(\mathbf{x}, \mathbf{z}|\theta)] + \frac{1}{2}\ln\left(\left|\Sigma^{-1}\right|\right) \tag{6}$$

where $\mathbf{x}$ is the latent trajectory, $\mathbf{z}$ represents the data measurements, and θ represents the network parameters. During EM training, the parameters over the joint factor $\phi(\mathbf{x}, \mathbf{z}|\theta)$ are split up into motion prior factors and measurement factors. For the motion prior factors, Burnett et al. apply a white-noise-on-acceleration prior presented by Anderson et al. [110], and the measurements factors are calculated using the learned keypoints, weights, and descriptors from the U-Net architecture. In the E-step of the GEM, the model parameters θ are hold fixed while optimizing Equation (6) for the posterior distribution $q(\mathbf{x})$, and during the M-step the posterior distribution is hold fixed while optimizing Equation (6) for the network parameters θ. A notable departure from the methodology proposed in [87] was the utilization of a moving window strategy. This approach involved incorporating more than just two consecutive radar scans into the analysis.

Another method to perform odometry estimation using external sensors is through scene flow prediction, which involves predicting the three-dimensional motion field of dynamic scenes. Such a method captures both the spatial and temporal aspects of static and dynamic objects within a perception scan of the environment. This is usually performed using a cost volume layer [111] to aggregate information across disparities or disparities and optical flow, facilitating the refinement of predictions. This layer computes matching costs between points at different disparities, enabling any network to capture intricate details and improve the accuracy of scene flow predictions by considering the cost relationships across different dimensions.

Ding et al. [89] introduced RaFLOW, depicted in Figure 5, a deep learning network specifically crafted for estimating the flow scene of a vehicle through the use of 4D automotive radar detections. The method involves processing two consecutive point clouds within a Radar-Oriented Flow Estimation module. Subsequently, a Static Flow Refinement module, employing the Kabsch algorithm [108], is applied to estimate the vehicle's ego-motion. Leveraging the determined ego-motion, the model then determines the scene flow of dynamic detections. This method integrates the ego-motion as one loss where the objective is to determine the complete scene flow.

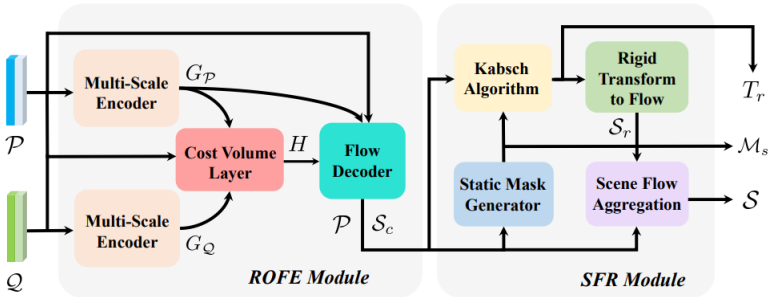

Figure 5. Overview of RaFlow [89]. Two point clouds, $\mathcal{P}$ and $\mathcal{Q}$ are inserted and first encoded using a multi-scale encoder. A cost volume layer is used with the point cloud and its features to approximate the scene flow of the entire scene in a flow decoder module. An additional static mask generator is used to filter out dynamic objects that could impact the ego-motion estimation by generating a mask $\mathcal{M}_s$. Finally, the Kabsch algorithm is applied to generate the ego-motion T_r. This transformation is used to further fine-tune the rest of the scene by predicting the final scene flow $\mathcal{S}$. Reproduced with permission from [89].

Lu et al. [90] presented 4DRO-Net, a novel approach for predicting radar odometry from 4D radar data. This technique employs a coarse-to-fine hierarchical optimization

strategy based on a sliding window estimation approach. Extracting separate features using PointNet-style feature encoders [112] from 4D radar data enables the fusion of features to generate both point and global features for each scan. The global features are instrumental in generating an initial pose estimate, while the point features contribute to a pose regression module utilizing a velocity-aware attentive cost volume layer between consecutive scaled point features. The corrected pose is then applied to the initial transformation assumption, yielding the final odometry between the two point clouds and resulting in ego-transformations at each investigated scale.

Finally, Almalioglu et al. [91] implemented Milli-RIO, a radar based odometry method based on point cloud allignment, A NDT [40] is used in an unscented Kalman-Filter (UKF) approach for point cloud alignment with a sophisticated point matching procedure. A RNN is implemented in the form of a LSTM layer as a motion model. Given a state of the system as

$$\boldsymbol{x}_t = [\boldsymbol{p}_t, \boldsymbol{q}_t, \boldsymbol{v}_t, \boldsymbol{b}_t^a]^T \tag{7}$$

with $\boldsymbol{x}_t$ being the state, $\boldsymbol{p}_t$ the position, $\boldsymbol{q}_t$ the rotation, $\boldsymbol{v}_t$ the velocity, and $\boldsymbol{b}_t^a$ bias of angular velocity at time t in Equation (7). Assuming a transition function $f(\cdot)$ for the sensor motion model and a constant bias for the bias of the angular velocity, the system state between consecutive scans is given by

$$\boldsymbol{x}_t = \left[\boldsymbol{p}_{t-1} + f(\boldsymbol{x}_{t-1}), \boldsymbol{q}_t \cdot \Delta\boldsymbol{q}, \boldsymbol{v}_{t-1}, \boldsymbol{b}_{t-1}^a\right]^T \tag{8}$$

A bidirectional-LSTM with 256 nodes is used to approximate the transition function $f(\boldsymbol{x}_{t-1})$ in Equation (8). A bidirectional LSTM is chosen, as it should caption the forward and backward dependencies of the motion model. The network is trained with ground truth trajectories and the predicted state is fed into the UKF.

As seen, deep learning methods for radar-based odometry focus on either spinning or automotive radar data, depending on the available representation. The processing steps and implementations differ depending on the representation.

For automotive radar, the data are generally represented as a sparse detection point cloud. The RCS and Doppler values are used as further features of each detection. The trend for automotive radars tends toward predicting the flow of all detections in a scene, and thus predicting the ego-motion via stationary objects. A variety of different layers, including PointNet-style feature enhancers, linear, warping, and cost-volume layers, are used to best approximate this transformation.

For spinning radar data, the focus is on identifying relevant features and keypoints from image-like radar scans. These models are able to operate in a polar or cartesian space, taking advantage of the denser representation of spinning radar. CNNs and transformer architectures are integrated for spatial feature extraction, while LSTM and GRU layers are implemented to establish a temporal connection between the spatial features.

For both radar setup configurations, the focus is still on finding a 2D transformation matrix $T \in SE(2)$. Most papers also utilize some masking module in which irrelevant matches or features are filter out. Only recent models try to utilize 4D automotive radars to find a 3D ego-transformation matrix $T \in SE(3)$.

5.2. Radar Based Loop Closure

Deep learning methods have gained significant prominence in the domain of loop closure detection. They are commonly integrated into existing localization frameworks, often in conjunction with components such as a Vector of Locally Aggregated Descriptors (VLAD) layer.

The VLAD layer, first proposed by Arandjelovic et al. [113], summarizes local features from inputs into a compact representation. Local features are initially extracted from a given set of input data. These extracted features are then organized into clusters using a clustering algorithm like k-means clustering. This clustering step generates a hidden dictionary of visually meaningful words, also referred to as "visual words". For each local

feature, a determination is made as to which visual word in the dictionary it is closest to. This assignment is made based on a chosen distance metric, such as the cosine similarity or Euclidean distance. After the assignments are made, the differences between the assigned visual words and the original features are accumulated and normalized. This aggregation process results in a compact descriptor that encodes the distribution of the differences across the visual words. The resulting dimension of the descriptor is calculated as the product of the number of visual words in the dictionary (C) and the initial dimension of the local features (D), yielding a descriptor of size $VLAD = C \times D$.

Building on this concept, the NetVLAD layer [114] extends the approach by introducing a learnable assignment stage prior to the VLAD aggregation. This involves incorporating a deep neural network that learns the encoding of the visual words and clusters. The training of the NetVLAD layer often employs a margin loss, which can be a triplet loss [115] or contrastive loss [116]. These loss functions guide the training process by encouraging meaningful representations in the learned descriptors space.

$$\mathcal{L}_{\text{triplet}}(A, P, N) = \max(0, d(A, P) - d(A, N) + \eta) \tag{9}$$

Equation (9) represents a standard triplet loss, where the variables have the following meanings: A refers to the anchor or query descriptor, P corresponds to positive descriptors that are physically close to the query location, and N denotes negative descriptors that are physically far away from the query location. η represents the margin. Positives are generally chosen as locations that are closer than 5 m within the query and negatives are further than 15 m away from the query location.

Cai et al. [92] fused a deep spatial–temporal encoder with a dynamic points removal scheme for pre-processing and a RCS based re-ranking score as post processing. The architecture is shown in Figure 6. The filtered representation is passed through a NetVLAD layer and optimized using a triplet loss to generate a unique scene descriptor. Finally, a RCS re-ranking is used to match scenes with similar RCS values, filtering out non-relevant stationary features, and reject places as potential loop closure detections if the histogram of RCS values are too dissimilar between a scan and a scan at a denoted keypose. The authors tested this method on the NuScenes dataset.

Suaftescu et al. [93] employed a VGG-16 backbone with a NetVLAD layer to extract a global invariant descriptor from a RA map captured by a spinning radar. The approach capitalizes on the polar representation of the scan, incorporating circular padding, anti-aliasing blurring, and azimuth-wise max-pooling techniques to achieve rotational invariance for the local descriptor. The network is optimized using a triplet loss, enhancing the separation between different classes within the data and aiming to extract robust features for radar-based place recognition. Building on this work, Gadd et al. [94] created a contrast-enhanced distance matrix between embeddings from places along reference and live trajectories. Place recognition is then performed through a closest neighborhood search based on the distance matrices of different places.

In a subsequent work, Gadd et al. [95,117] focused on unsupervised radar place recognition, employing an embedding learning approach to ensure that features of distinct instances are separated while features of augmented instances remain invariant. Using a VGG-19 as a frontend backbone, they extract local features that should be similar regardless of total augmentation. Similar to [94], difference matrices are created for localization based on a neighborhood search.

Yuan et al. [97] implemented a generative model, namely a variational autoencoder (VAE) [118], to create a place recognition method that incorporates the uncertainty to highlight the differences between positive sample pairs. The VAE is used to untangle the feature representation into variance of prediction-irrelevant uncertainty sources and semantic invariant descriptor. For negative pairs, the distance between the semantically invariant descriptors is maximized. For positive pairs, a Gaussian distribution is sampled with the found mean and variance of the VAE and added to the descriptor to take the uncertainty into consideration. This is especially relevant in highly noisy radar scans. The

uncertainty also allows for a consisting mapping maintenance, rejecting samples generated with high uncertainty compared to the already present map data.

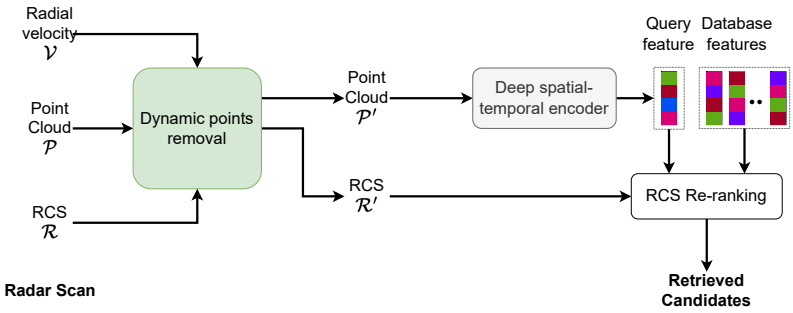

Figure 6. Similar place retrieval for Autoplace model [92]. Initially, a 2D point cloud, accompanied by its radial velocity and RCS score, serves as input. The first step involves filtering the point cloud to eliminate moving targets. The resultant filtered point cloud undergoes processing through a spatial–temporal encoder, comprising CNNs as spatial encoders and LSTMs as temporal encoders. The final representation is then fed through a NetVLAD layer, facilitating the extraction of essential features and spatial relationships. Subsequently, similar places, as determined by the NetVLAD layer, are further refined by re-ranking based on the Kullback–Leibler divergence of each RCS histogram. This comprehensive process enhances the model's ability to understand and differentiate scenes based on radar features.

Usuelli et al. [96] extended the work by Burnett et al. [88] by sampling from local descriptors with relevant keypoints to perform place recognition, shown in Figure 7. These descriptors are then processed through a NetVLAD layer adapted for point-wise global localization, originally implemented for a lidar approach [119]. They also used the cosine distance metric to evaluate the triplet loss, enhancing the discriminative power of the extracted features.

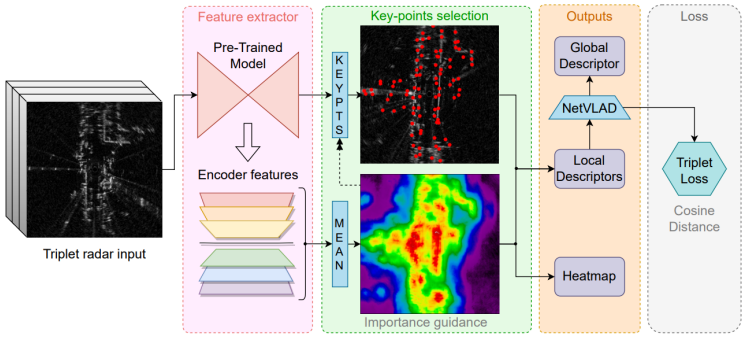

Figure 7. Training pipeline of RadarLCD for loop closure detection [96]. The model adopts the keypoint extraction and encoding structure from [87], with a modification in which the mean of the descriptor map is computed to generate an importance guidance map. This importance guidance map is then sampled at the keypoint locations and processed through a NetVLAD layer to generate a global descriptor. This pipeline ensures that keypoint information, coupled with an importance-guided global representation, is leveraged for effective perform radar-based loop closure detection. Reproduced with permission from [96].

Unlike prior works, Wang et al. [98] implemented RadarLoc that takes a single cartesian map from spinning radar scans as input and outputs a single pose $\boldsymbol{T}$ based on GT information. The architecture of the convolutional network comprises a self-attention masking module, followed by a DenseNet [120] feature extractor, and finally two separate multi-layer perception (MLP) layers for the untangled determination of the translational and rotational component. This configuration facilitates the extraction of a feature map

from the cartesian map input. Notably, a generalized max-pooling layer [121] is incorporated as a pooling mechanism, contributing to the extraction of global descriptors in the DenseNet module.

Adolfsson et al. [122] presented a sophisticated loop closure verification scheme for spinning radar data by employing two logistic regression classifiers. Although this method is not trained via deep learning, it is still relevant for state-of-the-art research. The process involves extracting hand-crafted CFEAR [36] features from a Cartesian Bird's Eye View (BEV) radar scan as global descriptors. These global descriptors are then utilized to identify loop closure candidates. An alignment measurement between the two scans is applied, extracting various features from the joined point clouds, including joint and separate entropy, radar peak detections, and a measure of overlap. These features serve as input for training a logistic classifier, reflecting the similarity of the two point clouds based on their overlap. Finally, this similarity, combined with the point clouds' similarity in odometry and global descriptor distance, is integrated into a logistic regression classifier scheme to determine a verified loop closure candidate. This comprehensive approach enhances the robustness of loop closure verification for spinning radar data.

The focus of radar based ego-localization can be categorized into two categories: methods that depend on training a model to generate unique local place descriptors, and methods that want to predict a global pose $T \in SE(3)$ from a single scan. The models generating a local descriptor are often trained using a NetVLAD layer in an End2End approach. To some extent, the same modules were integrated from the odometry methods into the place recognition methods, including CNNs and LSTMs for spatial and temporal feature extraction. Similar and dissimilar places are extracted from the training data to force a small distance metric of feature descriptors of similar places, and large differences between places that are physically far apart. This leads to a large dependency on the chosen similar and dissimilar distance criteria, as well as on the investigated dataset.

The other main method to perform localization is by predicting the ego-pose $T \in SE(3)$ directly from a local scan. These methods are far less researched in comparison to utilizing a BOW approach, because they do not rely on the usage of any map, and are thus very susceptible to over-fitting. However, these methods are more prevalent when adding an extra sensor to the system.

5.3. Fusion and Cascading Architectures

5.3.1. Vision Fusion

In many aspects, vision fusion has the greatest potential with radar data as the radar provides the cameras with much needed depth perception, while remaining relative cheap compared to a lidar for economical purposes. Simultaneously, vision also provides radar with much needed spatial context in "normal" lightning conditions [5,9].

Yu et al. [123] implemented a multi-modal deep localization scheme. The architecture contains two main branches; an image branch that encodes a camera image using CNN blocks and a point cloud branch with a PointNet [112] backbone. Fusion between the lidar and CNN branch is performed by spatial corresponding, point-wise image feature extraction, and feature fusion of each image encoder block and the resulting pointcloud representation block, which is then encoded into the pointcloud feature. Lastly, a NetVLAD layer is applied on the last fusion layer to train the network.

Almalioglu et al. [100] fuse camera data with range sensor data, namely radar or lidar, for a self-supervised geometry-aware multimodal ego-motion estimation network called GRAMME in all-weather scenarios. The network consists of various modules to estimate the ego-motion of a vehicle under various weather and lighting scenarios. A DepthNet U-Net produces depth images from camera inputs and a VisionNet predicts the ego-motion from consecutive camera frames. A MaskNet produces input masks from range inputs, while a RangeNet predicts the ego-motion based on the range sensor input. The RangeNet and VisionNet outputs are fused in a FusionNet module to predict the

ego-motion. Finally, the results from the DepthNet, MaskNet, and FusionNet are combined in a spatial transformer to train the model in a complete supervised fashion.

Zhuo et al. [124] implement a camera–radar fusion model for visual–radar odometry using 4D radar data called 4DRVO shown in Figure 8. The model fuses features using a multi-scale approach, where camera features are computed in a pyramid network and the radar features are computed in a pyramid style using an extension of PointNet++ [125] called Radar-PointNet++ where the additional RCS and Doppler information is incorporated besides the coordinate position of the detection. The visual and radar features are fused on multiple scales using the external camera–radar calibration matrix to project the radar features onto the feature map of the RGB image. The resulting projection and radar feature map is further fused using a deformable attention-based spatial cross-attention mechanism where the radar features identifies relevant objects in the RGB feature map. A cost volume layer is introduced to find associations between consecutive point clouds where a velocity-guided point confidence estimation is used to assign a confidence calibration to each radar detection, essentially learning to mask over non-stationary detections. MLPs are implemented to generate odometry estimation at each scale. Lastly, a Pose Warp Refinement Module is implemented to fuse the embeddings, confidence calibration, and pose estimation of the current and the next scale to further fine tune the pose estimation. The 4DRVO model is able to predict the odometry of the vehicle better than standalone radar or camera data and is even able to generate better results than a sole lidar sensor, demonstrating the importance of 4D radar data.

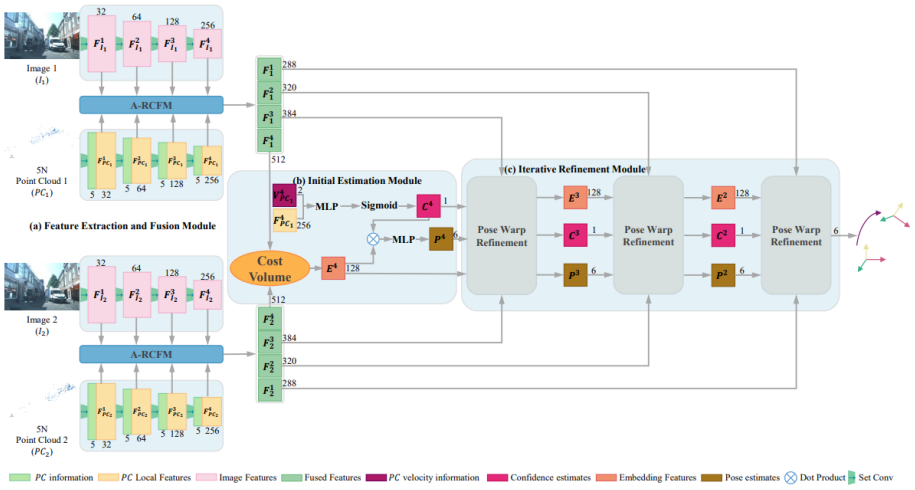

Figure 8. 4DRVO-Net [124] includes three main components. The first is a multi-scale extraction block for radar and visual data with an adaptive 4D radar–camera fusion module (A-RCFM) fusion block at each scale. Image features and 4D point features are extracted from the vision and 4D radar data respectively. The second is an initial pose estimation module that takes the two 4D point clouds at the coarsest scale and their features to produce a confidence score, embedding features and pose estimate. The final component is a pose refinement module that takes the confidences, embedding features, and predicted poses of a prior scale to progressively improve the estimates at each scale. Reproduced with permission from [124].

It is also possible to couple radars with other vision based sensors. Tang et al. [99,126] used satellite imagery to perform scan matching between an artificial and real radar scan. The authors implemented a generative-adversarial network (GAN) [127] to convert a satellite image to an artificial radar image. They used a conditional GAN, or cGAN, in order to condition the artificial radar image on a prior satellite image. The training consists of two parts. The first part involves training the cGAN to generate an artificial radar representation. The second part of the network consists of finding the translational and rotational components of the transformation matrix $T \in SE(2)$. Since no information about the altitude is present in either the spinning radar scan or the overhead image, no

information about the height transformation can be inferred. The network splits the search for the translation and rotation component by determining the angle of rotation through the biggest correlation in the Fourier domain between the original radar data and rotated versions of the artificial scan. Finally, linear and convolutional layers are used to determine the translational offset. This approach requires an approximately correct ego-pose in order to obtain a satellite image.

5.3.2. Lidar Fusion

Over recent years, many authors have implemented deep generative models to generate specific synthetic data from custom input data, where the generation of radar to lidar is frequently performed, as expensive lidar data can be exchanged for cheap radars with a generative model. The generative model not only performs translation between sensors, such as lidar to radar [128] or radar to lidar [101], but also translations between lidar representations in different weather conditions [129].

Based on generative modelling, Yin et al. [101] introduce a GAN to convert a spinning radar scan into a synthetic lidar map. Instead of sharing polar features, this network generates a synthetic lidar BEV image from a radar scan. The synthetic lidar output aims to remove the clutter in the radar scan while generating features similar to a lidar scan. The synthetic lidar scan is then used by the authors along with a prior map for odometry and ego-localization in a particle filter localization scheme. The odometry is estimated using point-to-plane ICP between consecutive lidar scans. Localization is performed using a metric based on number of matched points to the map.

Yin et al. [102] perform odometry on a prior lidar map with a known position using a differentiable Kalman-Filter approach shown in Figure 9. A cartesian 2D radar scan and a lidar submap are fed through several U-Net architectures to extract two separate feature embeddings from each input sensor. The lidar embedding is rotated by a range of angles $\delta\theta$ and translated by a range of possible solution factors Δx and Δy. The difference tensor between the radar embedding and each transformed lidar embedding is split up into patches and each patch is fed through another network to obtain a patch based difference. This difference is averaged and its softmin is taken to finally generate marginal probability distributions for each solution candidate, δx, δy and $\delta\theta$. The model is trained with three different loss functions. The first is a cross-entropy loss of the found marginal probability distributions using a one-hot encoding of the ground truth parameters. The second loss is the mean squared error between the ground truth parameters and the resulting estimated parameters from the marginal distributions and corresponding offset. Finally, the last loss is based on a differentiable Kalman-Filter to balance between the likelihood and a lower uncertainty in the posterior.

In a similar fashion to the approaches presented in [101,102], Lisus et al. [103] utilize deep learning to localize a radar scan on a lidar map. Specifically, a U-Net network is employed to extract a mask from the radar scan, similar to the methodology by [85]. Concurrently, a point cloud is extracted from the radar scan using the BFAR detector [130], with features from the mask sampled at the extracted points serving as weight values. A significant part of this method relies on a differential ICP backend, which matches the extracted points with a reference lidar map.

Sun et al. [131] implement a GAN network for moving object detection in lidar data. Although radar data is not used directly by the deep learning module, the Doppler velocity of the radar scan acts as a verification tool to ensure that dynamic objects have been accurately determined. Additionally, lidar data are utilized to filter ghost objects in the radar data.

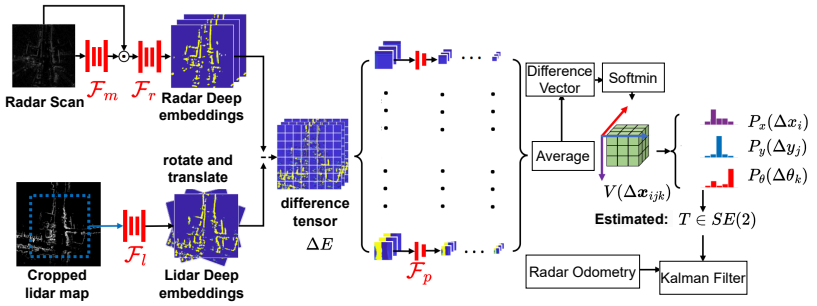

Figure 9. Overview of the RaLL architecture [102]. Three networks, denoted as $\mathcal{F}_m$, $\mathcal{F}_l$, and $\mathcal{F}_r$, collectively generate a difference tensor ΔE across a predefined search space. Subsequently, another network, $\mathcal{F}_p$, referred to as the patch network, is employed to construct a single-valued number for each patch and potential offset combination. This can be expressed in a single tensor $V(\Delta \mathbf{x}_{ijk})$. Probability distributions are then formed based on this tensor across each dimension. Finally, the pose offsets are generated from these distributions. The obtained pose offsets can be utilized within a Kalman-Filter approach, complemented by IMU data. This comprehensive architecture integrates multiple networks to effectively generate and utilize pose information for enhanced localization. Reproduced with permission from [102].

Yin et al. [104] conduct joint place recognition of radar and lidar scans by employing a shared encoder–decoder architecture for both the polar representation of a radar scan and a lidar submap. The output features are transformed into a BEV representation, and a 2D FFT is applied to generate a final vector descriptor. The network is trained using a joint training scheme, optimizing the loss with different sensors for each query sample (anchor, positive, negative).

Similar to the work by Yin et al. [104], the work by Nayak et al. [105] also leverages lidar submaps for radar-based localization on a lidar map. However, in contrast to [104], individual radar and lidar backbones are implemented using a Recurrent All-Pairs Field Transformer (RAFT) feature extractor [132] to generate a BEV feature representation and a unique descriptor for each sensor. This method utilizes a place recognition head for loop closure detection and a metric localization head to find the 3D pose $T \in SE(3)$ on the lidar map. During training, the place recognition head is trained with a triplet loss between the current radar descriptor and a positive and negative example of a lidar submap to enforce the similarity of descriptors between similar places recorded with different sensors. Interestingly, the application of a final NetVLAD layer did not produce better localization results than the final CNN layers. The encoded positive lidar submap is further used in the metric localization head to predict the complete flow in the radar scene. Akin to [89], this method also utilizes the correlation volume between the two BEV encodings as input for a GRU to predict the scene flow. Utilizing this flow and the localization head, the ego-pose of the system in the lidar map can be inferred from a single radar scan.

Ding et al. [106] implement a cross-modal model using 4D automotive radar sensors, a lidar sensor, camera images, and IMU odometry sensors for a supervised scene flow approach. The primary objective is to predict the complete scene flow of a scan, encompassing the rigid ego-motion and motion of other dynamic objects. Relevant stationary points of consecutive scans and corresponding weights are estimated from 4D radar point clouds processed through a backbone network consisting of *set conv* layers [133], an initial flow estimation head, and a motion segmentation head. Similar to [89], an SVD decomposition is applied to the found coordinates to derive a rigid transformation matrix, which is trained using the transformation found with the IMU sensor.

5.3.3. Cascading Architectures

Next, we discuss two exemplary methods that employ cascading techniques involving the utilization of results from other models, such as the annotation of radar points during object detection and labeling in a sensor fusion approach. This is to motivate the usage of

deep learning networks, e.g., as a pre-processing network that can assist in filtering out irrelevant detections with a non-learning based matching approach.

The work conducted by Isele et al. [107] integrates semantic segmentation results from other sensors to categorize radar detections into different classes. Subsequently, the segmented radar detections are filtered to retain only those belonging to relevant classes. The annotated radar point clouds are then subjected to NDT scan matching for alignment. Notably, deep learning is not directly applied to the radar data; instead, deep learning serves to enhance the radar features by incorporating information from other sensors. This approach showcases the synergistic integration of information from diverse sensors, leveraging deep learning to enrich the radar feature set with insights derived from semantic segmentation results. The cascading methodology enhances the discriminate power of radar data, ultimately contributing to improved matching accuracy in scenarios where collaborative sensor information is available.

Cheng et al. [134] use a GAN architecture to create synthetic pointclouds from a RD map. The CNN-based generator creates a mask for RD matrix, and the CNN-based discriminator is trained to differentiate between generator masks and Ground Truth masks created from lidar and synchronous radar data. The resulting synthetic point cloud can be used in object detection task or tracking, and in a SLAM approach for odometry estimation or ego-localization.

In fusion architectures, radar data is commonly utilized to generate denser synthetic lidar representations through techniques like GAN or VAE for odometry and ego-localization. Such approaches enable the use of synthetic data for odometry and localization methods tailored for denser lidar representations. However, generative networks operate on pre-processed radar representations, like point clouds or scan representations. This preprocessing may result in the loss of relevant information from the raw representation, replaced by synthetic data, potentially leading to inaccurate results.

5.4. Evaluation

In this subsection, we compare the introduced methods based on their radar setup, i.e. between automotive radar and spinning radar setups, for the odometry and ego-localization task. We try to compare these methods both in terms of how well they perform, but also try to highlight the importance of utilized hardware by analyzing the inference speed and device used to test the model. We also present relevant classical approaches for comparison to see how the deep learning models perform in comparison to established methods. The data come directly from the papers which are cited.

Finding a common ground for a fair comparison among the mentioned methods poses considerable challenges. To ensure a fair evaluation, methods should ideally be tested on the same dataset and under identical conditions. However, achieving this is complicated due to the inherent differences in datasets and conditions across various tasks, particularly when comparing methods designed for spinning radar sensors with those tailored for automotive radar datasets and vice versa.

Moreover, assessing inference speed is crucial for both odometry and localization tasks, especially in real-time applications like autonomous driving. The computation time of the model should ideally not be the primary bottleneck; instead, the focus should be on the data gathering speed of the perception sensor, such as radar data generation. This implies that fair comparisons require testing odometry or localization models on identical hardware.

However, practical challenges hinder the attainment of this ideal scenario. Data generation speeds can vary not only between different types of sensors (e.g., automotive vs. spinning) but even among sensors of the same type. For automotive sensors, additional preprocessing steps may introduce significant differences in data generation speeds between differently configured sensors or models. Standardizing to a single sensor is also impractical, as different applications demand varying sensor specifications, leading to differences in resolution, accuracy, and price.

Furthermore, the computational hardware of a system can constrain model choices. Implementing a large model may not be feasible if the system lacks the capability to support modern GPUs due to space or price limitations. Choosing an appropriate model that aligns with the environment of the system adds another layer of complexity, making the comparison of similar methods across different environments challenging. Despite these challenges, efforts to establish standardized evaluation metrics and practices will contribute to more meaningful comparisons in the field.

5.4.1. Odometry

In point-matching or pose estimation evaluations, determining the translational and rotational components of the pose difference between predicted and true poses is a fundamental aspect. The Absolute Trajectory Error (ATE) serves as a common metric, representing the average error for both translation and rotation components. Typically measured in meters (m) for translation and degrees (deg) for rotation, the ATE is calculated as follows for a pair of poses:

$$E_{tra} = ||\hat{t} - t||_2 \quad E_{rot} = ||\hat{R}R^T - \mathbf{I}||_2 \tag{10}$$

In Equation (10), $\hat{t}$ and t are the predicted and true translational components, and $\hat{R}$ and R are the predicted and true rotational components, respectively. The ATE combines these individual components into a single measure of translational and rotational error.

However, in the context of odometry estimation, where the global drift over a certain spatial extent is often more critical than small pose differences between consecutive frames, the ATE may not be the most appropriate metric. If a network learns to predict the transformation between two scans, it is more relevant to know how wrong the network is after some spatial or temporal extent than to know how the network performs in between consecutive scans. This is relevant especially in slow-moving systems with high data generation frequencies. An example would be the Oxford radar dataset, which operates at 4 Hz. Small pose differences may not accurately capture the global drift over a significant range in such cases.

To address this concern, the KITTI evaluation benchmark [135] introduces a metric that averages the relative position and rotation error over sub-sequences of various lengths (e.g., 100 m, 200 m, ..., 800 m). The resulting translational and rotational ATE is reported as a percentage offset for the translational component and in degrees per 1000 m for the rotational component. This provides a more comprehensive evaluation metric, considering the performance of the system over the extended distances and offering a metric relevant for odometry tasks.

Spinning Radar

The methods presented in Table 4 showcase four distinct approaches evaluated using the same data and same metric; the Oxford radar dataset using the KITTI metric on spinning radar. Fast-MbyM [86] demonstrates the fastest inference speed, achieving a notable speed-up compared to the MbyM method with only a marginal sacrifice in accuracy. Among the deep learning models, RaLL [102] stands out as the top performer, although it requires additional LiDAR data during training for effective conversion between radar and LiDAR representations using a GAN.

Observing Table 4, it becomes evident that correlation models generally outperform keypoint extraction and matching methods. Notably, methods like [85–88] and have the added advantage of using learned keypoints and weights for both odometry estimation and localization, rendering them more versatile in an end-to-end approach. However, it is worth mentioning that methods by [87,88] exhibit higher inference times and require more substantial GPU support. Despite this, given the 4Hz rotation frequency of the spinning radar in the Oxford robot dataset, all methods remain applicable in real-time scenarios.

Table 4. List of deep learning methods that have been employed for radar-based odometry and tested on the mean Oxford robot test dataset. Performance evaluations are derived from the respective citations. Models marked with (*) indicate training involving additional LiDAR data, while those marked with (**) need additional camera data. The methods marked with (***) represent state-of-the-art classical radar odometry methods. Best performing models are highlighted with **bold**.

Method	KITII (%/$\frac{\text{deg}}{\text{km}}$)	Inference Speed (s)	Device
MbyM [85]	1.16/**3.0**	0.0747	Nvidia Titan Xp GPU (Santa Clara, CA, USA)
Fast-MbyM [86]	2.00/6.3	**0.0063**	Jetson GPU (Nvidia, Santa Clara, CA, USA)
RaLL * [102]	**1.09**/3.7	0.243	Nvidia Titan X GPU (Santa Clara, CA, USA)
UnderTheRadar [87]	2.06/6.7	0.034	-
HERO [88]	1.95/6.5	0.180	Nvidia Tesla V100 GPU (Santa Clara, CA, USA)
GRAMME ** [88]	1.98/5.1	-	NVIDIA GTX 1080Ti (Santa Clara, CA, USA)
CFEAR-3 *** [136]	1.09/3.6	0.167	i7-6700K CPU (Intel, Santa Clara, CA, USA)

In comparison to CFEAR [90], deep learning methods demonstrate comparable performance in terms of accuracy and computational time. However, it is important to note that the CFEAR evaluation was conducted on an Intel i7-8700k CPU, while the other models required GPU support for evaluation.

Automotive Radar

Table 5 presents a collection of odometry methods tailored for automotive radar and evaluated on the same set of the View-of-Delft dataset. Automotive radar data, known for their sparsity, are typically utilized for object detection and ego-localization rather than odometry. Given the inherent challenges of sparse radar point clouds, the focus in this context is on odometry estimation, particularly leveraging the 4D nature of radar point clouds in the View-of-Delft dataset. Notably, all models in Table 5 underwent evaluation on a single NVIDIA 2080Ti GPU, ensuring a more reliable inference comparison than the previous comparison.

Table 5. List of deep learning methods assessed for radar-based odometry on the View-of-Delft (VoD) dataset. The methods are evaluated on VoD Seq. 4, and the data are sourced from [90]. Notably, (*) incorporates Camera, Lidar, and IMU data during training, providing a multimodal learning approach. Additionally, (**) represents a classical method based on the normal distribution transform. The evaluations are conducted on a single NVIDIA 2080Ti GPU. Best performing models are highlighted with **bold**.

Method	ATE (m/deg)	Inference Speed (s)
RaFlow [89]	0.07/0.45	0.0363
CMFlow * [106]	**0.05**/0.10	0.0304
4DRO-Net [90]	0.08/**0.07**	**0.0108**
NDT **	0.56 / 1.52	0.0010

Among the radar-only setups, RaFlow [89] and 4DRO-Net [90] demonstrate competitive performance. Interestingly, CMFlow [106], which integrates additional information such as camera, LiDAR, and IMU data in its training setup, showcases comparable results. Notably, the radar-only methods, especially 4DRO-Net [90], exhibit noteworthy performance in ATE and inference speed even when compared to the CMFlow setup. Moreover, the deep learning methods consistently outperform the classical NDT transform, indicating significant advancements in performance despite the computational efficiency of the classic approach, which operates without the need for a GPU during odometry prediction.

5.4.2. Ego-Localization

Ego-localization methods necessitate distinct evaluation criteria compared to odometry estimation. While Equation (10) can be used for predicting scan matching in detected loop closures, it is inadequate for assessing the ability of a model to recognize already visited

locations. Typically, a scene descriptor, akin to the (Net)VLAD descriptors, is generated to encapsulate the place and facilitate place recognition.

One evaluation approach involves a nearest-neighborhood (NN) search, determining the closest neighbors of the observed descriptor in a descriptor database using a defined distance metric (e.g., that specified in the NetVLAD loss of Equation (9)). Metrics like Recall@1, F1 score, and mean Average Precision (mAP) are commonly employed for assessing consistent loop closures through precision–recall curves. However, a challenge lies in defining the number of pre-defined keyposes in the dictionary, where a smaller distance between keyposes can lead to a worse metric performance as the dictionary size grows, without necessarily indicating an overall inferior model.

Similar to radar odometry, place recognition models are predominantly evaluated using spinning radar data. However, unlike odometry methods, GPU requirements are less critical for these approaches, as they do not need to run continuously. Once the location on a map has been determined, an odometry network can be utilized for pose refinement. Additionally, if ego-localization is performed on a prior map, having both on the same device is computationally efficient, eliminating the need for data transfer. The choice between performing localization on an external cluster or using on-board hardware presents distinct challenges, each with its own set of considerations. In either case, the model should exhibit sufficient speed to enable re-localization in the event of global drift or when a recognized scene is detected.

Table 6 presents the place recognition performance of various deep learning models designed for spinning radar, assessed using the Oxford dataset and evaluated with metrics such as Recall@1, F1, and mAP. For comparison, the non-learning-based descriptor method RaPlace [137] is included. However, challenges arise in the evaluation criteria due to variations in positive boundary definitions, with [94] using a stricter 10 m boundary compared to the less strict 25 m boundaries employed by other models.

Table 6. List of deep learning methods applied to spinning radar-based localization on the Oxford radar dataset. Notably, (*) has a positive boundary of 10 m, while the other methods have a boundary of 25 m. Additionally, (**) incorporates an additional LiDAR submap during training. Best performing models are highlighted with **bold**.

Method	Recall@1	mAP	F1	Inference (s)	Device
Kidnapped radar [93]	90.49	0.75	0.7	-	-
Look around you * [94]	-	0.53	0.53	-	-
Contrastive Learning [95]	85.39	0.95	0.89	-	-
Off the Radar [97]	93.67	**0.99**	**0.95**	-	-
RadarLCD [96]	-	0.94	-	-	-
Radar-to-Lidar ** [104]	93.18	-	-	-	-
RaLF ** [105]	**97.0**	-	-	-	2 NVIDIA TITAN-X GPU (Santa Clara, CA, USA)
RaPlace * [137]	78.0	-	-	0.24	16-Core Intel i7-11700 CPU (Santa Clara, CA, USA)

Different models excel depending on the evaluation metric. RaLF [105] achieves the highest Recall@1 metric, surpassing pure radar models and a similar model [104] that utilizes a LiDAR submap during training. Off the Radar [97] demonstrates superior performance in terms of mAP and F1 score, outperforming even [104]. Notably, the model with the best Recall@1 fails to provide additional metrics, highlighting the need for a uniform localization metric across models.

Table 6 also reveals a lack of information regarding inference rates for the models, with RaPlace being the only exception. Additionally, there is limited detail on the hardware used for running the models. Considering that the classical approach closely matches the spinning radar frequency (4 Hz), it is crucial to assess whether a model can operate effectively in an online environment.

Some models include a fine pose-matching layer, often utilizing a lidar submap, where the transformation matrix between a pose and a keypose is estimated after place recognition. The evaluation metric for these models aligns with odometry assessments. However, in

this context, pose differences are more significant than those close to the identity matrix, reflecting a larger average displacement between poses and making translation and rotation ATE more meaningful.

Table 7 indicates that models incorporating scan-matching techniques, such as RaLL [102] and RaLF [105], outperform RadarLoc, emphasizing the significance of including additional information beyond the scan data. This underscores the importance of implementing a place-recognition head, as relying solely on a single 6DOF head may not be sufficient for effective localization, particularly in large and possibly repeating environments.

Table 7. List of deep learning methods applied for radar based metric localization on the Oxford Radar dataset. (*) only predicts a single 6DOF transformation matrix with a displacement error.

Method	δ x (m)	δ y (m)	δ θ (deg)
RadarLoc * [98]	2.53	2.53	2.81
RaLL [102]	1.04	**0.69**	1.26
RaLF [105]	**0.97**	**0.69**	**1.15**

In conclusion, Table 8 highlights the disparity between an automotive radar-only localization models and a state-of-the-art approach employing camera and LiDAR sensors. The latter outperforms the radar-only pipeline by approximately 7% points for Recall@1 and 4% points for the F1 score. While the state-of-the-art approach showcases superior performance, the results still underscore the reliability of the radar-only model. Further investigation could explore the potential benefits of sensor fusion models, which have demonstrated their utility in odometry estimation.

Table 8. Automotive localization comparison between Autoplace [92] and a vision-LiDAR localization model [138] marked with (*) on the NuScenes dataset.

Method	Recall@1	F1
Autoplace [92]	90.45	0.95
LCPR * [138]	97.34	0.99

6. Discussion and Future Research

Deep learning techniques have gradually made their way into radar-based odometry and localization methods, primarily serving as spatial and temporal feature encoders. These approaches often employ CNNs for spatial encoding and temporal feature encoders in the form of RNN networks, such as LSTMs and GRUs, to gain temporal information about the same object in different scenes.

Regardless of task, the modules and layers implemented differ in the configuration of the radar. Automotive radar configurations focus on PointNet and PointNet++ feature encoders to work with a point cloud representation. These feature encoders are chosen due to their simple architecture, enabling a fast inference time due to the shared MLP structure of PointNet. PointNet and its modifications have also been applied extensively for various state-of-the-art methods for point cloud processing in other fields [139–144]. Spinning radar focuses on CNNs to encode the representation. Similar to the occlusion issue in computer vision, most models try to generate a mask or weight function to deal with the noisy and potentially faulty radar data [145].

Compared to other radar perception tasks, the adoption of deep learning in this context has been somewhat limited. Although state-of-the-art deep learning is used, the number of papers published for radar odometry and localization is far below the number of pages published for object detection and segmentation for radars [5,9]. Similar to [107], the results from segmentation papers [145–147] can be used to mask out irrelevant detections and areas from scans that do not contribute to the odometry or localization of the system. In a complete pipeline, segmentation, classification, odometry, and ego-localization should be performed in an end-to-end approach, where the results from one module are

integrated, e.g. as a mask, for another module. In a similar fashion, semantic networks for unsupervised classification have been successfully applied in the fields of computer vision [148] and natural language processing [149].

Despite the availability of processed radar representations like BEV images or point clouds, which are more intuitive for odometry and localization tasks compared to pre-CFAR representations, there is a limited application of deep learning approaches to solve radar-based odometry or localization directly from less processed representations. While the ADC or RAD representation may contain irrelevant information, such as clutter or dynamic objects, deep neural networks have the capability to learn to mask these regions and focus on ego-motion and unique global descriptors based on these representations. As masking is a huge part in spinning radar odometry, it is surprising that no similar methods are applied to pre-CFAR representations for automotive radar. Such an approach could potentially overcome the issue of discarding relevant objects during pre-processing, offering an advantage in scenarios where radar data are available as RAD tensors or as ADC signals, such as in K-Radar or RadIAL datasets.

In contrast, a notable trend in other radar perception tasks involves a shift away from heavily pre-processed 2D/3D data (e.g., BEV or point cloud representations) and toward less processed representations such as ADC [150] or RAD [74]. The rationale behind this trend is to equip deep learning models to discern relevant features within the input space and mask out irrelevant elements. While this trend is observable in radar-based odometry, particularly in processing RA maps for spinning radars, the automotive radar field has not yet followed a similar trajectory, as it utilizes the post CFAR detections for PointNet feature encoding. Moreover, while the usage of 4D radar data is increasing for tasks like object detection and segmentation, their adoption in odometry and localization methods remains limited to a handful of approaches.

Another trend observed is the dominance of spinning radar configurations in the investigated sensor setups. For odometry and localization tasks, the number of papers utilizing spinning radar data outweighs those using automotive radar. However, given the lower cost and easier integration of automotive radars into existing vehicles, there is a potential for a greater research focus on such radar sensors.

Furthermore, we observed that many models evaluating on the same dataset follow vastly different evaluation criteria, both in terms of their evaluation method and the inference speed of the model. Unlike object detection and segmentation, where NMS and IOU scores are widely accepted, odometry and place recognition tasks lack standardized evaluation criteria, leading to variations based on external parameters such as the true pose parameter in determining a true recognized place.

In the realm of ego*localization systems, most papers mention that the system performs faster than the recording frequency. However, in real-time applications, additional factors need consideration, such as data transfer during place recognition and pose refinement. For odometry applications, the hardware components used differ significantly among papers, making it challenging to compare results directly. For deep learning approaches, the inference time is especially important in regard to optimization and fine-tuning [151,152]. If a model does not work in real time on a platform, there are various steps that can be taken, e.g., lowering the amount of parameters in the network or changing the hardware of the system. It is also possible that some hardware in the system will be upgraded, or some properties of the radar will change. Then, the model needs to quickly adapt to these changes, and it needs to be controlled to ensure that the network is still compatible with the new properties of the data, signaling the importance of transfer learning.

Additionally, in contrast to radar-based odometry and localization, the broader field of sensor perception is witnessing a focus on advanced deep learning techniques. These include variations in transformers [153,154], the utilization of various point feature encoder [112,155] architectures, and the incorporation of rotational invariance into deep learning modules [156,157] through model architectures instead of augmentations during

training. These advancements are influencing not only general radar perception but also odometry and localization models across other modalities.

Author Contributions: Conceptualization: M.B. and A.P.; writing—original draft preparation, M.B.; supervision, A.P. and T.M.; Review, A.P. and T.M.; project administration, A.P. and T.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bavle, H.; Sanchez-Lopez, J.L.; Cimarelli, C.; Tourani, A.; Voos, H. From slam to situational awareness: Challenges and survey. *Sensors* **2023**, *23*, 4849. [CrossRef]
2. Ryde, J.; Hillier, N. Performance of laser and radar ranging devices in adverse environmental conditions. *J. Field Robot.* **2009**, *26*, 712–727. [CrossRef]
3. He, G.; Yuan, X.; Zhuang, Y.; Hu, H. An integrated GNSS/LiDAR-SLAM pose estimation framework for large-scale map building in partially GNSS-denied environments. *IEEE Trans. Instrum. Meas.* **2020**, *70*, 1–9. [CrossRef]
4. Yao, S.; Guan, R.; Huang, X.; Li, Z.; Sha, X.; Yue, Y.; Lim, E.G.; Seo, H.; Man, K.L.; Zhu, X.; et al. Radar-camera fusion for object detection and semantic segmentation in autonomous driving: A comprehensive review. *arXiv* **2023**, arXiv:2304.10410.
5. Harlow, K.; Jang, H.; Barfoot, T.D.; Kim, A.; Heckman, C. A New Wave in Robotics: Survey on Recent mmWave Radar Applications in Robotics. *arXiv* **2023**, arXiv:2305.01135.
6. Zhang, J.; Xie, Y.; Ling, L.; Folkesson, J. A Fully-automatic Side-scan Sonar SLAM Framework. *arXiv* **2023**, arXiv:2304.01854.
7. Steiniger, Y.; Kraus, D.; Meisen, T. Survey on deep learning based computer vision for sonar imagery. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105–157. [CrossRef]
8. Chen, C.; Wang, B.; Lu, C.X.; Trigoni, N.; Markham, A. A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence. *arXiv* **2020**, arXiv:2006.12567.
9. Zhou, Y.; Liu, L.; Zhao, H.; López-Benítez, M.; Yu, L.; Yue, Y. Towards deep radar perception for autonomous driving: Datasets, methods, and challenges. *Sensors* **2022**, *22*, 4208. [CrossRef]
10. Visentin, T.; Hasch, J.; Zwick, T. Analysis of multipath and DOA detection using a fully polarimetric automotive radar. *Int. J. Microw. Wirel. Technol.* **2018**, *10*, 570–577. [CrossRef]
11. Durrant-Whyte, H.; Bailey, T. Simultaneous localization and mapping: Part I. *IEEE Robot. Autom. Mag.* **2006**, *13*, 99–110. [CrossRef]
12. Bailey, T.; Durrant-Whyte, H. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robot. Autom. Mag.* **2006**, *13*, 108–117. [CrossRef]
13. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv* **2019**, arXiv:1910.13461.
14. Voulodimos, A.; Doulamis, N.; Doulamis, A.; Protopapadakis, E. Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.* **2018**, *2018*, 7068349. [CrossRef]
15. Sarlin, P.E.; DeTone, D.; Malisiewicz, T.; Rabinovich, A. Superglue: Learning feature matching with graph neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2020; pp. 4938–4947.
16. Zhu, A.Z.; Liu, W.; Wang, Z.; Kumar, V.; Daniilidis, K. Robustness meets deep learning: An end-to-end hybrid pipeline for unsupervised learning of egomotion. *arXiv* **2018**, arXiv:1812.08351.
17. Liu, X.; Chen, S.W.; Aditya, S.; Sivakumar, N.; Dcunha, S.; Qu, C.; Taylor, C.J.; Das, J.; Kumar, V. Robust fruit counting: Combining deep learning, tracking, and structure from motion. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1045–1052.
18. Sarlin, P.E.; Cadena, C.; Siegwart, R.; Dymczyk, M. From coarse to fine: Robust hierarchical localization at large scale. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12716–12725.
19. Filatov, A.; Rykov, A.; Murashkin, V. Any motion detector: Learning class-agnostic scene dynamics from a sequence of lidar point clouds. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Pisctaway, NJ, USA, 2020; pp. 9498–9504.
20. Mohammadi, M.E.; Watson, D.P.; Wood, R.L. Deep learning-based damage detection from aerial SfM point clouds. *Drones* **2019**, *3*, 68. [CrossRef]

21. Im, J.U.; Ki, S.W.; Won, J.H. Omni Point: 3D LiDAR-based Feature Extraction Method for Place Recognition and Point Registration. *IEEE Trans. Intell. Veh.* **2024**, 1–18. [CrossRef]
22. O'Shea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458.
23. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef]
24. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
25. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]
26. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
27. Zhang, Y.; Wu, Y.; Tong, K.; Chen, H.; Yuan, Y. Review of Visual Simultaneous Localization and Mapping Based on Deep Learning. *Remote Sens.* **2023**, *15*, 2740. [CrossRef]
28. Mokssit, S.; Licea, D.B.; Guermah, B.; Ghogho, M. Deep Learning Techniques for Visual SLAM: A Survey. *IEEE Access* **2023**, *11*, 20026–20050. [CrossRef]
29. Saleem, H.; Malekian, R.; Munir, H. Neural Network-Based Recent Research Developments in SLAM for Autonomous Ground Vehicles: A Review. *IEEE Sens. J.* **2023**, *23*, 13829–13858. [CrossRef]
30. Bilik, I. Comparative Analysis of Radar and Lidar Technologies for Automotive Applications. *IEEE Intell. Transp. Syst. Mag.* **2022**, *15*, 244–269. [CrossRef]
31. Grisetti, G.; Kümmerle, R.; Stachniss, C.; Burgard, W. A tutorial on graph-based SLAM. *IEEE Intell. Transp. Syst. Mag.* **2010**, *2*, 31–43. [CrossRef]
32. Wang, Z.; Li, W.; Shen, Y.; Cai, B. 4-D SLAM: An efficient dynamic bayes network-based approach for dynamic scene understanding. *IEEE Access* **2020**, *8*, 219996–220014. [CrossRef]
33. Ru, X.; Gu, N.; Shang, H.; Zhang, H. MEMS inertial sensor calibration technology: Current status and future trends. *Micromachines* **2022**, *13*, 879. [CrossRef] [PubMed]
34. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; IEEE: Pisctaway, NJ, USA, 2011; pp. 2564–2571.
35. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
36. Adolfsson, D.; Magnusson, M.; Alhashimi, A.; Lilienthal, A.J.; Andreasson, H. Cfear radarodometry-conservative filtering for efficient and accurate radar odometry. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; IEEE: Pisctaway, NJ, USA, 2021; pp. 5462–5469.
37. Segal, A.; Haehnel, D.; Thrun, S. Generalized-icp. In Proceedings of the Robotics: Science and Systems, Seattle, WA, USA, 28 June–1 July 1; Volume 2, p. 435.
38. Zhang, J.; Yao, Y.; Deng, B. Fast and robust iterative closest point. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 3450–3466. [CrossRef]
39. Derpanis, K.G. Overview of the RANSAC Algorithm. *Image Rochester NY* **2010**, *4*, 2–3.
40. Biber, P.; Straßer, W. The normal distributions transform: A new approach to laser scan matching. In Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453), Las Vegas, NV, USA, 27 October–1 November 2003; IEEE: Piscataway, NJ, USA, 2003; Volume 3, pp. 2743–2748.
41. Kellner, D.; Barjenbruch, M.; Klappstein, J.; Dickmann, J.; Dietmayer, K. Instantaneous ego-motion estimation using doppler radar. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, The Netherlands, 6–9 October 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 869–874.
42. Kellner, D.; Barjenbruch, M.; Klappstein, J.; Dickmann, J.; Dietmayer, K. Instantaneous ego-motion estimation using multiple Doppler radars. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1592–1597.
43. Cui, Y.; Chen, X.; Zhang, Y.; Dong, J.; Wu, Q.; Zhu, F. Bow3d: Bag of words for real-time loop closing in 3d lidar slam. *IEEE Robot. Autom. Lett.* **2022**, *8*, 2828–2835. [CrossRef]
44. Louback da Silva Lubanco, D.; Schlechter, T.; Pichler-Scheder, M.; Kastl, C. Survey on Radar Odometry. In Proceedings of the Computer Aided Systems Theory–EUROCAST 2022: 18th International Conference, Las Palmas de Gran Canaria, Spain, 20–25 February 2022; Revised Selected Papers; Springer: Berlin/Heidelberg, Germany, 2023; pp. 619–625.
45. Geng, Z.; Yan, H.; Zhang, J.; Zhu, D. Deep-learning for radar: A survey. *IEEE Access* **2021**, *9*, 141800–141818. [CrossRef]
46. Kuutti, S.; Fallah, S.; Katsaros, K.; Dianati, M.; Mccullough, F.; Mouzakitis, A. A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications. *IEEE Internet Things J.* **2018**, *5*, 829–846. [CrossRef]
47. Arshad, S.; Kim, G.W. Role of deep learning in loop closure detection for visual and lidar slam: A survey. *Sensors* **2021**, *21*, 1243. [CrossRef] [PubMed]
48. Roy, P.; Chowdhury, C. A survey of machine learning techniques for indoor localization and navigation systems. *J. Intell. Robot. Syst.* **2021**, *101*, 63. [CrossRef]

49. Yin, P.; Zhao, S.; Cisneros, I.; Abuduweili, A.; Huang, G.; Milford, M.; Liu, C.; Choset, H.; Scherer, S. General Place Recognition Survey: Towards the Real-world Autonomy Age. *arXiv* **2022**, arXiv:2209.04497.
50. Xu, X.; Zhang, L.; Yang, J.; Cao, C.; Wang, W.; Ran, Y.; Tan, Z.; Luo, M. A review of multi-sensor fusion slam systems based on 3D LIDAR. *Remote Sens.* **2022**, *14*, 2835. [CrossRef]
51. Chghaf, M.; Rodriguez, S.; Ouardi, A.E. Camera, LiDAR and multi-modal SLAM systems for autonomous ground vehicles: A survey. *J. Intell. Robot. Syst.* **2022**, *105*, 2. [CrossRef]
52. Huang, L. Review on LiDAR-based SLAM techniques. In Proceedings of the 2021 International Conference on Signal Processing and Machine Learning (CONF-SPML), Beijing, China, 18–20 August 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 163–168.
53. Khan, M.U.; Zaidi, S.A.A.; Ishtiaq, A.; Bukhari, S.U.R.; Samer, S.; Farman, A. A comparative survey of lidar-slam and lidar based sensor technologies. In Proceedings of the 2021 Mohammad Ali Jinnah University International Conference on Computing (MAJICC), Karachi, Pakistan, 15–17 July 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–8.
54. Chen, W.; Shang, G.; Ji, A.; Zhou, C.; Wang, X.; Xu, C.; Li, Z.; Hu, K. An overview on visual slam: From tradition to semantic. *Remote Sens.* **2022**, *14*, 3010. [CrossRef]
55. Lowry, S.; Sünderhauf, N.; Newman, P.; Leonard, J.J.; Cox, D.; Corke, P.; Milford, M.J. Visual place recognition: A survey. *IEEE Trans. Robot.* **2015**, *32*, 1–19. [CrossRef]
56. Wang, K.; Ma, S.; Chen, J.; Ren, F.; Lu, J. Approaches, challenges, and applications for deep visual odometry: Toward complicated and emerging areas. *IEEE Trans. Cogn. Dev. Syst.* **2020**, *14*, 35–49. [CrossRef]
57. Duan, C.; Junginger, S.; Huang, J.; Jin, K.; Thurow, K. Deep learning for visual SLAM in transportation robotics: A review. *Transp. Saf. Environ.* **2019**, *1*, 177–184. [CrossRef]
58. Placed, J.A.; Strader, J.; Carrillo, H.; Atanasov, N.; Indelman, V.; Carlone, L.; Castellanos, J.A. A survey on active simultaneous localization and mapping: State of the art and new frontiers. *IEEE Trans. Robot.* **2023**, *39*, 1686–1705. [CrossRef]
59. Favorskaya, M.N. Deep Learning for Visual SLAM: The State-of-the-Art and Future Trends. *Electronics* **2023**, *12*, 2006. [CrossRef]
60. Zeng, F.; Wang, C.; Ge, S.S. A survey on visual navigation for artificial agents with deep reinforcement learning. *IEEE Access* **2020**, *8*, 135426–135442. [CrossRef]
61. Zeng, J.; Wang, D.; Chen, P. A Survey on Transformers for Point Cloud Processing: An Updated Overview. *IEEE Access* **2022**, *10*, 86510–86527. [CrossRef]
62. Yi, Z. A Survey of Radar Perception—Datasets, Methods and Applications. Available online: https://github.com/ZHOUYI1023/awesome-radar-perception (accessed on 28 February 2024).
63. Barnes, D.; Gadd, M.; Murcutt, P.; Newman, P.; Posner, I. The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 6433–6438.
64. Burnett, K.; Yoon, D.J.; Wu, Y.; Li, A.Z.; Zhang, H.; Lu, S.; Qian, J.; Tseng, W.K.; Lambert, A.; Leung, K.Y.; et al. Boreas: A multi-season autonomous driving dataset. *Int. J. Robot. Res.* **2023**, *42*, 33–42. [CrossRef]
65. Kim, G.; Park, Y.S.; Cho, Y.; Jeong, J.; Kim, A. Mulran: Multimodal range dataset for urban place recognition. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 6246–6253.
66. Sheeny, M.; De Pellegrin, E.; Mukherjee, S.; Ahrabian, A.; Wang, S.; Wallace, A. RADIATE: A radar dataset for automotive perception in bad weather. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–7.
67. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11621–11631.
68. Palffy, A.; Pool, E.; Baratam, S.; Kooij, J.F.; Gavrila, D.M. Multi-class road user detection with 3+ 1D radar in the View-of-Delft dataset. *IEEE Robot. Autom. Lett.* **2022**, *7*, 4961–4968. [CrossRef]
69. Kramer, A.; Harlow, K.; Williams, C.; Heckman, C. ColoRadar: The direct 3D millimeter wave radar dataset. *Int. J. Robot. Res.* **2022**, *41*, 351–360. [CrossRef]
70. Cheng, Y.; Jiang, M.; Zhu, J.; Liu, Y. Are we ready for unmanned surface vehicles in inland waterways? The usvinland multisensor dataset and benchmark. *IEEE Robot. Autom. Lett.* **2021**, *6*, 3964–3970. [CrossRef]
71. Mostajabi, M.; Wang, C.M.; Ranjan, D.; Hsyu, G. High-resolution radar dataset for semi-supervised learning of dynamic objects. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 13–19 June 2020; pp. 100–101.
72. Yan, Z.; Sun, L.; Krajník, T.; Ruichek, Y. EU long-term dataset with multiple sensors for autonomous driving. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 10697–10704.
73. Meyer, M.; Kuschk, G. Automotive radar dataset for deep learning based 3d object detection. In Proceedings of the 2019 16th European Radar Conference (EuRAD), Paris, France, 2–4 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 129–132.
74. Paek, D.H.; Kong, S.H.; Wijaya, K.T. K-Radar: 4D radar object detection for autonomous driving in various weather conditions. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 3819–3829.

75. Rebut, J.; Ouaknine, A.; Malik, W.; Pérez, P. Raw high-definition radar for multi-task learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 17021–17030.
76. Goppelt, M.; Blöcher, H.L.; Menzel, W. Automotive radar–investigation of mutual interference mechanisms. *Adv. Radio Sci.* **2010**, *8*, 55–60. [CrossRef]
77. Xu, Z. Bi-Level l 1 Optimization-Based Interference Reduction for Millimeter Wave Radars. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 728–738. [CrossRef]
78. Li, X.; Zhang, H.; Chen, W. 4D Radar-based Pose Graph SLAM with Ego-velocity Pre-integration Factor. *IEEE Robot. Autom. Lett.* **2023**, *8*, 5124–5131. [CrossRef]
79. Zhang, J.; Zhuge, H.; Wu, Z.; Peng, G.; Wen, M.; Liu, Y.; Wang, D. 4DRadarSLAM: A 4D Imaging Radar SLAM System for Large-scale Environments based on Pose Graph Optimization. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), Londin, UK, 29 May–2 June 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 8333–8340.
80. Khan, K.; Rehman, S.U.; Aziz, K.; Fong, S.; Sarasvady, S. DBSCAN: Past, present and future. In Proceedings of the Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014), Chennai, India, 17–19 February 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 232–238.
81. Jose, E.; Adams, M.D. Relative radar cross section based feature identification with millimeter wave radar for outdoor slam. In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566), Sendai, Japan, 28 September–2 October 2004; IEEE: Piscataway, NJ, USA, 2004; Volume 1, pp. 425–430.
82. Jose, E.; Adams, M.; Mullane, J.S.; Patrikalakis, N.M. Predicting millimeter wave radar spectra for autonomous navigation. *IEEE Sens. J.* **2010**, *10*, 960–971. [CrossRef]
83. Wang, S.; Clark, R.; Wen, H.; Trigoni, N. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 2043–2050.
84. Aldera, R.; De Martini, D.; Gadd, M.; Newman, P. Fast radar motion estimation with a learnt focus of attention using weak supervision. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1190–1196.
85. Barnes, D.; Weston, R.; Posner, I. Masking by moving: Learning distraction-free radar odometry from pose information. *arXiv* **2019**, arXiv:1909.03752.
86. Weston, R.; Gadd, M.; De Martini, D.; Newman, P.; Posner, I. Fast-MbyM: Leveraging Translational Invariance of the Fourier Transform for Efficient and Accurate Radar Odometry. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 2186–2192.
87. Barnes, D.; Posner, I. Under the radar: Learning to predict robust keypoints for odometry estimation and metric localisation in radar. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 9484–9490.
88. Burnett, K.; Yoon, D.J.; Schoellig, A.P.; Barfoot, T.D. Radar odometry combining probabilistic estimation and unsupervised feature learning. *arXiv* **2021**, arXiv:2105.14152.
89. Ding, F.; Pan, Z.; Deng, Y.; Deng, J.; Lu, C.X. Self-supervised scene flow estimation with 4-d automotive radar. *IEEE Robot. Autom. Lett.* **2022**, *7*, 8233–8240. [CrossRef]
90. Lu, S.; Zhuo, G.; Xiong, L.; Zhu, X.; Zheng, L.; He, Z.; Zhou, M.; Lu, X.; Bai, J. Efficient Deep-Learning 4D Automotive Radar Odometry Method. *IEEE Trans. Intell. Veh.* **2023**, *9*, 879–892. [CrossRef]
91. Almalioglu, Y.; Turan, M.; Lu, C.X.; Trigoni, N.; Markham, A. Milli-RIO: Ego-motion estimation with low-cost millimetre-wave radar. *IEEE Sens. J.* **2020**, *21*, 3314–3323. [CrossRef]
92. Cai, K.; Wang, B.; Lu, C.X. Autoplace: Robust place recognition with low-cost single-chip automotive radar. *arXiv* **2021**, arXiv:2109.08652.
93. Săftescu, Ş.; Gadd, M.; De Martini, D.; Barnes, D.; Newman, P. Kidnapped radar: Topological radar localisation using rotationally-invariant metric learning. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 4358–4364.
94. Gadd, M.; De Martini, D.; Newman, P. Look around you: Sequence-based radar place recognition with learned rotational invariance. In Proceedings of the 2020 IEEE/ION Position, Location and Navigation Symposium (PLANS), Portland, OR, USA, 20–23 April 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 270–276.
95. Gadd, M.; De Martini, D.; Newman, P. Contrastive learning for unsupervised radar place recognition. In Proceedings of the 2021 20th International Conference on Advanced Robotics (ICAR), Ljubljana, Slovenia, 6–10 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 344–349.
96. Usuelli, M.; Frosi, M.; Cudrano, P.; Mentasti, S.; Matteucci, M. RadarLCD: Learnable Radar-based Loop Closure Detection Pipeline. *arXiv* **2023**, arXiv:2309.07094.
97. Yuan, J.; Newman, P.; Gadd, M. Off the Radar: Uncertainty-Aware Radar Place Recognition with Introspective Querying and Map Maintenance. *arXiv* **2023**, arXiv:2306.12556.
98. Wang, W.; de Gusmao, P.P.; Yang, B.; Markham, A.; Trigoni, N. Radarloc: Learning to relocalize in fmcw radar. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xian, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 5809–5815.

99. Tang, T.Y.; De Martini, D.; Barnes, D.; Newman, P. Rsl-net: Localising in satellite images from a radar on the ground. *IEEE Robot. Autom. Lett.* **2020**, *5*, 1087–1094. [CrossRef]
100. Almalioglu, Y.; Turan, M.; Trigoni, N.; Markham, A. Deep learning-based robust positioning for all-weather autonomous driving. *Nat. Mach. Intell.* **2022**, *4*, 749–760. [CrossRef] [PubMed]
101. Yin, H.; Wang, Y.; Wu, J.; Xiong, R. Radar style transfer for metric robot localisation on lidar maps. *CAAI Trans. Intell. Technol.* **2023**, *8*, 139–148. [CrossRef]
102. Yin, H.; Chen, R.; Wang, Y.; Xiong, R. RaLL: End-to-end Radar Localization on Lidar Map Using Differentiable Measurement Model. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 6737–6750. [CrossRef]
103. Lisus, D.; Laconte, J.; Burnett, K.; Barfoot, T.D. Pointing the Way: Refining Radar-Lidar Localization Using Learned ICP Weights. *arXiv* **2023**, arXiv:2309.08731.
104. Yin, H.; Xu, X.; Wang, Y.; Xiong, R. Radar-to-lidar: Heterogeneous place recognition via joint learning. *Front. Robot. AI* **2021**, *8*, 661199. [CrossRef]
105. Nayak, A.; Cattaneo, D.; Valada, A. RaLF: Flow-based Global and Metric Radar Localization in LiDAR Maps. *arXiv* **2023**, arXiv:2309.09875.
106. Ding, F.; Palffy, A.; Gavrila, D.M.; Lu, C.X. Hidden gems: 4D radar scene flow learning using cross-modal supervision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 9340–9349.
107. Isele, S.T.; Haas-Fickinger, F.; Zöllner, J.M. SERALOC: SLAM on semantically annotated radar point-clouds. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 1–19 September 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 2917–2924.
108. Kabsch, W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. Sect. Cryst. Phys. Diffr. Theor. Gen. Crystallogr.* **1976**, *32*, 922–923. [CrossRef]
109. Barnes, D.; Maddern, W.; Pascoe, G.; Posner, I. Driven to distraction: Self-supervised distractor learning for robust monocular visual odometry in urban environments. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1894–1900.
110. Anderson, S.; Barfoot, T.D. Full STEAM ahead: Exactly sparse Gaussian process regression for batch continuous-time trajectory estimation on SE (3). In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 157–164.
111. Sun, D.; Yang, X.; Liu, M.Y.; Kautz, J. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8934–8943.
112. 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, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
113. Arandjelovic, R.; Zisserman, A. All about VLAD. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 1578–1585.
114. 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, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 5297–5307.
115. Hoffer, E.; Ailon, N. Deep metric learning using triplet network. In Proceedings of the Similarity-Based Pattern Recognition: Third International Workshop, SIMBAD 2015, Copenhagen, Denmark, 2–14 October 2015; Proceedings 3; Springer: Berlin/Heidelberg, Germany, 2015; pp. 84–92.
116. Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; Krishnan, D. Supervised contrastive learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 18661–18673.
117. Gadd, M.; De Martini, D.; Newman, P. Unsupervised place recognition with deep embedding learning over radar videos. *arXiv* **2021**, arXiv:2106.067031.
118. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. *arXiv* **2013**, arXiv:1312.6114.
119. 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, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4470–4479.
120. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
121. Radenović, F.; Tolias, G.; Chum, O. Fine-tuning CNN image retrieval with no human annotation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 1655–1668. [CrossRef] [PubMed]
122. Adolfsson, D.; Karlsson, M.; Kubelka, V.; Magnusson, M.; Andreasson, H. TBV Radar SLAM–trust but verify loop candidates. *IEEE Robot. Autom. Lett.* **2023**, *8*, 3613–3620. [CrossRef]
123. Yu, X.; Zhou, B.; Chang, Z.; Qian, K.; Fang, F. MMDF: Multi-Modal Deep Feature Based Place Recognition of Mobile Robots with Applications on Cross-Scene Navigation. *IEEE Robot. Autom. Lett.* **2022**, *7*, 6742–6749. [CrossRef]
124. Zhuo, G.; Lu, S.; Zhou, H.; Zheng, L.; Xiong, L. 4DRVO-Net: Deep 4D Radar-Visual Odometry Using Multi-Modal and Multi-Scale Adaptive Fusion. *arXiv* **2023**, arXiv:2308.06573.
125. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.

126. Tang, T.Y.; De Martini, D.; Wu, S.; Newman, P. Self-supervised localisation between range sensors and overhead imagery. *arXiv* **2020**, arXiv:2006.02108.
127. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
128. Wang, L.; Goldluecke, B.; Anklam, C. L2R GAN: LiDAR-to-radar translation. In Proceedings of the Asian Conference on Computer Vision, Kyoto, Japan, 30 November 2020.
129. Lee, J.; Shiotsuka, D.; Nishimori, T.; Nakao, K.; Kamijo, S. Gan-based lidar translation between sunny and adverse weather for autonomous driving and driving simulation. *Sensors* **2022**, *22*, 5287. [CrossRef] [PubMed]
130. Alhashimi, A.; Adolfsson, D.; Magnusson, M.; Andreasson, H.; Lilienthal, A.J. Bfar-bounded false alarm rate detector for improved radar odometry estimation. *arXiv* **2021**, arXiv:2109.09669.
131. Sun, B.; Gao, S.; Zi, H.; Wu, Q. GAN based simultaneous localization and mapping framework in dynamic environment. *J. King Saud-Univ. Sci.* **2022**, *34*, 102298. [CrossRef]
132. Teed, Z.; Deng, J. Raft: Recurrent all-pairs field transforms for optical flow. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part II 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 402–419.
133. Liu, X.; Qi, C.R.; Guibas, L.J. Flownet3D: Learning scene flow in 3D point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 529–537.
134. Cheng, Y.; Su, J.; Jiang, M.; Liu, Y. A novel radar point cloud generation method for robot environment perception. *IEEE Trans. Robot.* **2022**, *38*, 3754–3773. [CrossRef]
135. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
136. Adolfsson, D.; Magnusson, M.; Alhashimi, A.; Lilienthal, A.J.; Andreasson, H. Lidar-Level Localization with Radar? The CFEAR Approach to Accurate, Fast, and Robust Large-Scale Radar Odometry in Diverse Environments. *IEEE Trans. Robot.* **2022**, *39*, 1476–1495. [CrossRef]
137. Jang, H.; Jung, M.; Kim, A. RaPlace: Place Recognition for Imaging Radar using Radon Transform and Mutable Threshold. *arXiv* **2023**, arXiv:2307.04321.
138. Zhou, Z.; Xu, J.; Xiong, G.; Ma, J. LCPR: A Multi-Scale Attention-Based LiDAR-Camera Fusion Network for Place Recognition. *arXiv* **2023**, arXiv:2311.03198.
139. Zhang, F.; Fang, J.; Wah, B.; Torr, P. Deep fusionnet for point cloud semantic segmentation. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part XXIV 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 644–663.
140. Zhang, Y.; Zhou, Z.; David, P.; Yue, X.; Xi, Z.; Gong, B.; Foroosh, H. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 9601–9610.
141. Chen, X.; Jiang, K.; Zhu, Y.; Wang, X.; Yun, T. Individual tree crown segmentation directly from UAV-borne LiDAR data using the PointNet of deep learning. *Forests* **2021**, *12*, 131. [CrossRef]
142. Cao, P.; Chen, H.; Zhang, Y.; Wang, G. Multi-view frustum pointnet for object detection in autonomous driving. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 3896–3899.
143. Wang, L.; Chen, T.; Anklam, C.; Goldluecke, B. High dimensional frustum pointnet for 3D object detection from camera, lidar, and radar. In Proceedings of the 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, USA, 19 October–13 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1621–1628.
144. Bi, J.; Wei, H.; Zhang, G.; Yang, K.; Song, Z. DyFusion: Cross-Attention 3D Object Detection with Dynamic Fusion. *IEEE Lat. Am. Trans.* **2024**, *22*, 106–112. [CrossRef]
145. Li, F.; Li, X.; Liu, Q.; Li, Z. Occlusion handling and multi-scale pedestrian detection based on deep learning: A review. *IEEE Access* **2022**, *10*, 19937–19957. [CrossRef]
146. Orr, I.; Cohen, M.; Zalevsky, Z. High-resolution radar road segmentation using weakly supervised learning. *Nat. Mach. Intell.* **2021**, *3*, 239–246. [CrossRef]
147. Ouaknine, A.; Newson, A.; Pérez, P.; Tupin, F.; Rebut, J. Multi-view radar semantic segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 15671–15680.
148. Zhang, C.; Cheng, J.; Tian, Q. Unsupervised and semi-supervised image classification with weak semantic consistency. *IEEE Trans. Multimed.* **2019**, *21*, 2482–2491. [CrossRef]
149. Pawar, A.; Mago, V. Challenging the boundaries of unsupervised learning for semantic similarity. *IEEE Access* **2019**, *7*, 16291–16308. [CrossRef]
150. Zhang, A.; Nowruzi, F.E.; Laganiere, R. Raddet: Range-azimuth-doppler based radar object detection for dynamic road users. In Proceedings of the 2021 18th Conference on Robots and Vision (CRV), Burnaby, BC, Canada, 26–28 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 95–102.
151. Goodfellow, I.J.; Vinyals, O.; Saxe, A.M. Qualitatively characterizing neural network optimization problems. *arXiv* **2014**, arXiv:1412.6544.

152. Subramanian, M.; Shanmugavadivel, K.; Nandhini, P. On fine-tuning deep learning models using transfer learning and hyper-parameters optimization for disease identification in maize leaves. *Neural Comput. Appl.* **2022**, *34*, 13951–13968. [CrossRef]
153. Dalbah, Y.; Lahoud, J.; Cholakkal, H. RadarFormer: Lightweight and accurate real-time radar object detection model. In Proceedings of the Scandinavian Conference on Image Analysis, Sirkka, Finland, 18–21 April 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 341–358.
154. Lee, D.; Nam, H.; Shim, D.H. ELiOT: End-to-end Lidar Odometry using Transformer Framework. *arXiv* **2023**, arXiv:2307.11998.
155. Yoon, D.J.; Zhang, H.; Gridseth, M.; Thomas, H.; Barfoot, T.D. Unsupervised learning of lidar features for use ina probabilistic trajectory estimator. *IEEE Robot. Autom. Lett.* **2021**, *6*, 2130–2138. [CrossRef]
156. Kim, G.; Kim, A. Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 4802–4809.
157. Kim, G.; Choi, S.; Kim, A. Scan context++: Structural place recognition robust to rotation and lateral variations in urban environments. *IEEE Trans. Robot.* **2021**, *38*, 1856–1874. [CrossRef]