9

Radar SLAM

Martin Magnusson, Christoffer Heckman, Henrik Andreasson, Ayoung Kim,
Timothy Barfoot, Michael Kaess, and Paul Newman

In this chapter, we explore the use of radar (RAdio Detection and Ranging) in SLAM. Compared to cameras and lidar, radar is somewhat undersubscribed. However, due to its ability to work in poor visibility, at long range, and to natively produce velocity information, its popularity is on the rise. We begin by discussing the types of radar sensor typically used in robotics, their unique sensing principles, and some of the challenges that come along with radar. We then discuss radar filtering, radar odometry, place recognition, and finally SLAM; Figure 9.1 shows how these pieces fit together. We conclude with an outlook on the use of radar moving forwards.

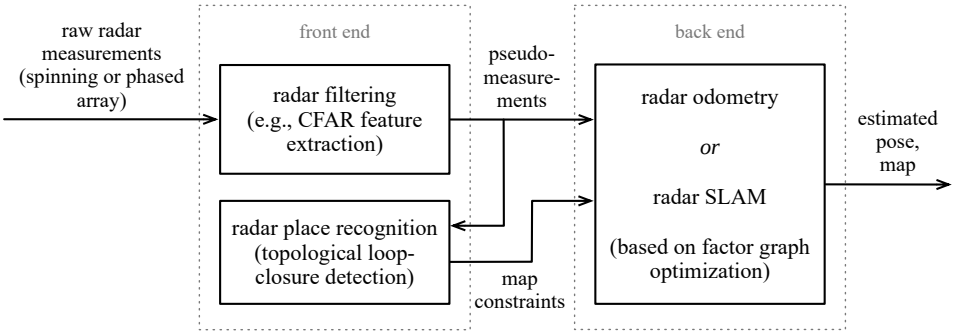

Figure 9.1 The flow of information in radar-based SLAM follows the same pattern as other SLAM systems with the details of each process being slightly altered to suit the specifics of radar.

9.1 Introduction to Radar

9.1.1 Sensor Types

We detail in the following section two of the most common radar types that are encountered in robotics: *spinning radar* and *system-on-a-chip (SoC) radar* (see Figure 9.2). Each has its strengths and drawbacks. They differ mainly in how they are

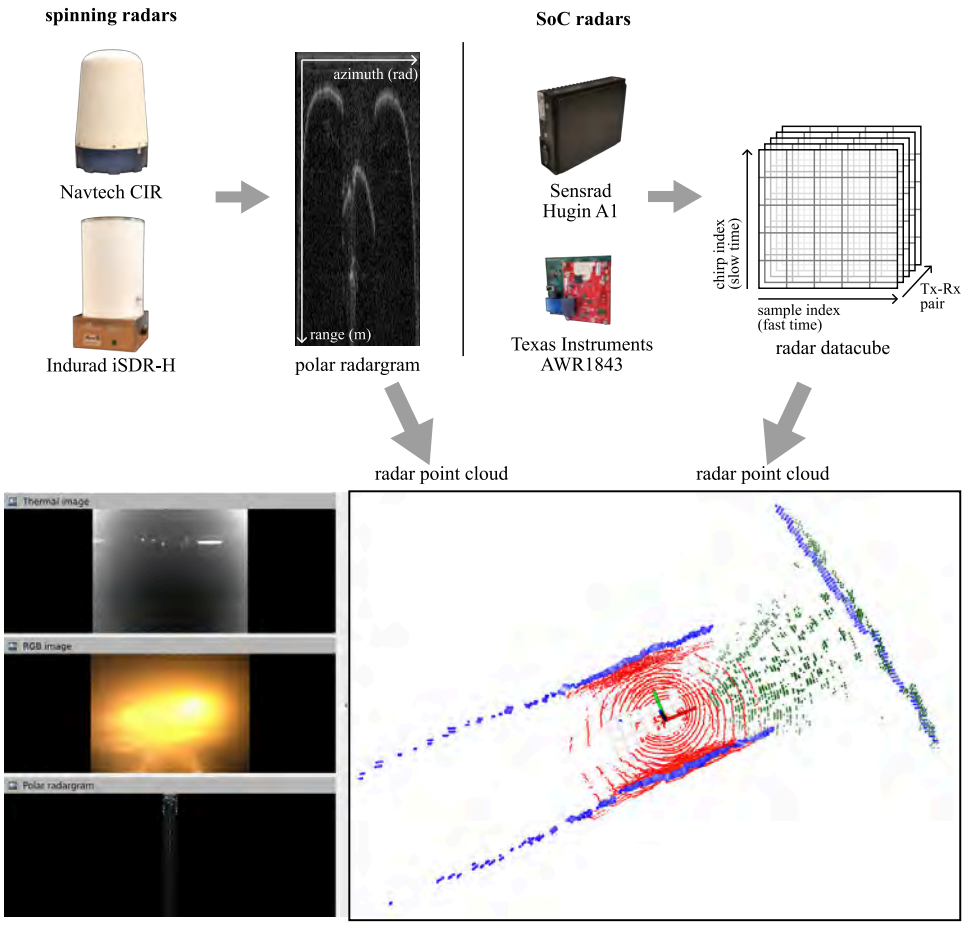

Figure 9.2 The two main categories of radars used in robotics are *spinning* (top left) and phased-array *SoC* (top right). Some examples of each are shown along with the main data product produced by each type of sensor. Often these raw data products are subsequently turned into a sparse point cloud using radar filtering (bottom). The bottom right part shows three point clouds from a tunnel with smoke. Red: LiDAR point cloud with much limited range due to the smoke. Blue: point cloud from 2D spinning radar (tunnel walls in all directions are clearly visible also at a distance). Green: 3D point cloud from SoC radar (walls and ground surface in front of the vehicle are visible). The bottom left part shows thermal and visual/RGB image data from the same scene, and the polar radargram from which the blue point cloud is extracted.

'actuated' with spinning radar mechanically rotating a single antenna and SoC using multiple antennas whose signals are combined to deduce the angles and ranges of objects reflecting the transmitted signals.

9.1.1.1 Spinning Radar

Utilizing a rotating radar sensor, *spinning radar* – sometimes referred to as scanning radar or imaging radar – crafts precise polar representations of its surroundings, exemplified in Figure 9.2. The main data product produced is a *polar radargram*. Imaging radars are distinguished by their ability to detect objects at distances exceeding 100 meters. In some modes, the velocity of those objects can also be determined by exploiting the Doppler effect.

In contrast to LiDAR systems, spinning radars are limited to providing data on a two-dimensional plane, lacking the capacity to measure the elevation of detected objects. This limitation persists even though reflections might originate from various elevations within the antenna's vertical beamwidth. Additionally, because their operational mechanism involves transmitting and receiving a single pulse for each antenna angle, these radars sometimes do not offer velocity data, which requires multiple pulses for calculation. More recent work uses the signal from multiple neighbouring azimuths to recover velocity.

Constructing a mechanically spinning 3D radar with multiple vertical beams, similar to currently widespread 3D LiDAR sensors, is not practically feasible due to the much larger size of the antennas and focusing mechanisms compared to laser diodes.

9.1.1.2 SoC Radar

System-on-a-chip (SoC) radars integrate processing units within a minimal set of chips, which are either directly mounted on patch antennas (arrays of transmitters / receivers) or incorporated into the printed circuit board itself. SoC radars are characterized by their lightweight design and reduced power requirements compared to spinning radars, thanks to their integrated architecture and lack of moving parts. The performance in terms of accuracy and resolution for SoC radars hinges on the design of the antenna array and the proprietary processing techniques manufacturers employ to integrate measurements from multiple antennas. The primary output produced is a *radar datacube*.

SoC returns are typically mapped in spherical coordinates by azimuth, elevation, and range. Given that radial velocity adds another dimension, these radars are often referred to as 3+1D or 4D. Conversely, systems with limited or absent vertical resolution, due to a scarcity of vertically aligned antennas, are denoted as 2D array radar systems, producing 2+1D data products.

9.1.2 Radar Sensing Principles

In this section, we will discuss basic operations, antennas and datatypes, and challenges as they relate to robotics applications. We outline each in the following

sections and designate how radar differentiates itself from other rangefinding sensors.

MmWave (millimeter-wave) radar is designated by radar systems whose electromagnetic wavelengths are between 1–10 mm with frequency ranging from 30–300 GHz. Within this range, most radar sensors operate in the 76–81 GHz segment of the spectrum due to automotive applications such as ADAS systems having spectrum carve-outs in this range.

9.1.2.1 Radar Cross Section

The process of mmWave radar involves emitting electromagnetic pulses that travel until they meet objects, bouncing back towards the radar. The Radar Cross Section (RCS), influenced by an object's material composition, size, and shape, plays a crucial role in determining how strongly each object reflects these electromagnetic pulses. Essentially, the RCS represents the size of a hypothetical sphere that would reflect the same amount of energy as the target object. Thus, larger and more solid structures such as vehicles and thick concrete walls exhibit a higher RCS compared to smaller objects or pedestrians, which present a lower RCS.

The term *radar intensity* refers to the strength of the radar's echo from an object, which is a function of the radar's transmitted power and the object's RCS. In essence, this intensity is chiefly determined by the radar's emitted power and the RCS of the encountered target. Literature highlights that this measure of intensity has been crucial for extracting semantic details about objects or aiding in navigation, as stronger echoes tend to be associated with distinctive and easily recognizable features in the environment [1235]. The strength of the radar signal is influenced by several additional factors, including the type of antenna used, the characteristics of the electromagnetic pulse emitted, and the antenna's ability to detect returns from objects.

9.1.2.2 FMCW Ranging

A radar comprises at least one transmitting antenna (TX) and one receiving antenna (RX).[1] In the context of FMCW radar, the TX antenna's role is to emit an RF pulse that steadily increases in frequency, known as a *chirp*. This is often called *sawtooth modulation*; we will also discuss *triangular modulation*, another chirp-based technique, later on. These chirps bounce off objects in the environment, and the RX antenna captures the echoes (see Figure 9.3). Upon receiving a chirp, the signal is amplified and then mixed with, or subtracted from, the original TX chirp to generate an Intermediate Frequency (IF), which is also a signal. The mixing process results in an IF that is a sine wave of constant frequency f_0, due to the identical slope of both the transmitted and received signals. The performance of the radar system, including its range and velocity detection capabilities, is affected by

[1] Although the TX and RX antennas can occasionally be the same unit, most SoC sensors opt to separate them, allowing for the incorporation of multiple TX and RX antennas.

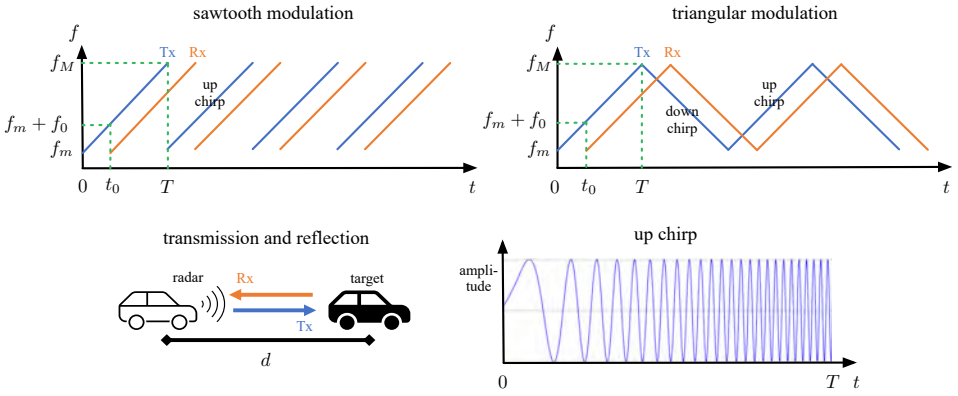

Figure 9.3 A frequency modulated continuous-wave (FMCW) radar works by emitting *chirps* (waves of increasing or decreasing frequency) that are reflected off targets and then received. Range (and velocity) are determined by analyzing the frequency (and phase) shifts of the reflected signal compared to the transmitted one. Two common frequency modulation strategies are *sawtooth* and *triangular*; the advantage of the latter being that the Doppler frequency shift can be disentangled from the range shift.

various chirp parameters such as the bandwidth (the difference between the initial and final frequencies), the chirp slope, and the duration between chirps.

The main idea of this *frequency modulation* is to encode temporal information onto a continuous wave, enabling the execution of range calculations. This technique stands in contrast to *amplitude modulation*, where the precision of the returned signal depends exclusively on the frequency's bandwidth. This characteristic renders frequency modulation more resilient to issues such as signal-to-noise ratios and the RCS of targets, which might otherwise affect the clarity of individual returns. In the context of FMCW radar, where waves are interspersed with unique intervals between pulses in a train, the signals maintain coherence. This coherence allows for each chirp to be accurately associated with its originating transmit-receive pair and its position within the sequence, facilitating the correlation of signals in terms of both amplitude and phase. This represents a significant evolution from the older time-of-flight pulsed radar systems, which relied on incoherent amplitude measurements and required the expertise of skilled operators to sift through clutter and isolate significant signals.

In FMCW, the calculation of distance relies on the temporal gap between when a signal is sent and when its echo is received. Utilizing the speed of light, c, and the initial arrival time, t_0, the distance, d, to an object is computed as

$$d = \frac{c}{2} t_0. \tag{9.1}$$

The top-left portion of Figure 9.3 shows the region where an IF signal is generated

using green dashed lines. The difference between the transmitted and received signal is an IF signal. The IF signal is a sine wave $\sin(2\pi f_0 t + \phi_0)$ whose frequency is proportional to a constant f_0 that spans from t_0 to T that only depends on the distance to the target, and is offset in phase by ϕ_0.

The frequency, f_0, is defined as a function of the distance to the target, the duration of the transmitted chirp, T, and the bandwidth of the transmitted signal, $B = f_M - f_m$. The bandwidth and transmit time are related to the slope of the chirp, $S = B/T$, as

$$f_0 = \frac{2Bd}{Tc} = \frac{2Sd}{c} \quad \Rightarrow \quad d = \frac{cf_0}{2S}, \tag{9.2}$$

where we now have the distance, d, as a function of the (measured) intermediate frequency, f_0.

In this section, although the equations are presented within the frequency domain, the real-world capture of signals predominantly occurs through digital sampling. This sampling is done at a high rate, typically every 100 nanoseconds or less, using a high-frequency Analog-to-Digital Converter (ADC). Following this, the sampled data undergoes a sequence of Fast Fourier Transform (FFT) operations. These operations generate graphs depicting frequency and amplitude, from which signal peaks can be discerned. The identification of range frequencies is achieved by applying FFT to the data from a single chirp and its corresponding return signal. To calculate velocity frequencies, a series of FFTs are executed on multiple chirp and return sequences.

9.1.2.3 Determining Distance and Velocity with Sawtooth Modulation

Sawtooth frequency modulation (see Figure 9.3) is typically used with SoC radars and some spinning radars. The phase of the IF signal, ϕ_0, can be expressed as a function of the wavelength λ of the signal and the distance d:

$$\phi_0 = 2\pi f_m t_0 = \frac{4\pi d}{\lambda}. \tag{9.3}$$

While both the base frequency f_0 and phase ϕ_0 are functions of distance d, the phase is only valid for sufficiently small distance values and is subject to angle-wrapping. Thus this is typically not used for range estimation but to measure small changes Δd where the phase responds linearly in velocity estimation.

For radial velocity estimation (see Figure 9.4), at least two sequential up chirps are employed as shown in the top left of Figure 9.3. As the distance in time between each chirp in the sequence is small, on the order of 40 microseconds, the range measurement from both samples and thus the relative IF f_i and f_{i+1} are nearly identical. However, the IF signals will possess distinct phases. This phase disparity $\Delta\phi$ corresponds to a motion of the object. The estimated velocity v is determined

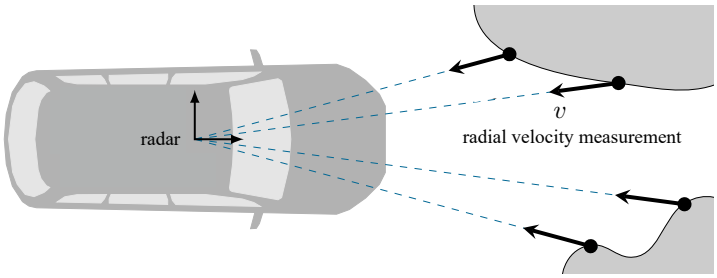

Figure 9.4 Radar is able to use the Doppler effect to measure the *radial* velocity between a unit and the scene it is imaging.

from the phase difference,

$$\Delta\phi = \frac{4\pi\Delta d}{\lambda} = \frac{4\pi v T}{\lambda}, \tag{9.4}$$

simplified to

$$v = \frac{\lambda\Delta\phi}{4\pi T}. \tag{9.5}$$

As velocity is a function of phase, we note the maximum detectable velocity is unambiguous for $|\Delta\phi| < \pi$. Thus $v_{\max} = \lambda/(4T)$ is a function of the wavelength of the signal and the time between chirps.

9.1.2.4 Determining Distance and Velocity with Triangular Modulation

Triangular frequency modulation (see Figure 9.3) can also be used; for example, it is sometimes used with spinning radars. In reality, when an intermediate frequency is derived from an up chirp, it comprises two components: (i) frequency shift due to the time of flight of the signal (discussed already), $f_{0,t}$, and (ii) the apparent frequency shift due to the *Doppler effect* if there is a relative velocity between the radar and the target, $f_{0,d}$:

$$f_{0,\text{up}} = f_{0,t} + f_{0,d}. \tag{9.6}$$

However, if we follow an up chirp with a down chirp that reflects off the same target, the sign of the temporal frequency shift will flip while that of the Doppler shift will not:

$$f_{0,\text{down}} = -f_{0,t} + f_{0,d}. \tag{9.7}$$

From these two equations we can solve for $f_{0,t}$ and $f_{0,d}$:

$$f_{0,t} = \frac{f_{0,\text{up}} - f_{0,\text{down}}}{2}, \quad f_{0,d} = \frac{f_{0,\text{up}} + f_{0,\text{down}}}{2}. \tag{9.8}$$

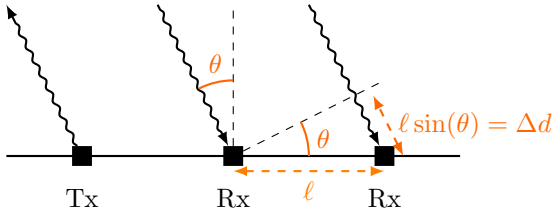

Figure 9.5 Angle of arrival estimation with phased-array radar. The measured difference in phase of the signal emitted by the TX transmitter antenna as received by two RX receiver antennas at a distance ℓ corresponds to the angle θ to the target.

Finally, from these two components, we can calculate the range and velocity according to

$$d = \frac{cf_{0,t}}{2S}, \quad v = \frac{cf_{0,d}}{2ST}. \tag{9.9}$$

Notably, the range calculation presented earlier in (9.2) is not corrected for the Doppler effect whereas this one is. The downside of using triangular modulation is an increase in latency since we now require slightly older data in the calculation of range and velocity. However, we do not need to work with the phase of the signal.

9.1.2.5 Determining Angle for SoC radar

For spinning radar, determining the angle to a target is trivial since the beam is focused in a single azimuth direction at each time. Angle estimation for SoC radar is slightly more complex but can be achieved with a similar phase difference calculation as above.

Given multiple receiver units separated by an interval ℓ, the distance disparity Δd emerges in reflections. The angle of arrival θ can be derived from the modification of (9.3) with the geometric relation $\Delta d = \ell \sin\theta$. The received signal must travel an extra distance $\ell \sin\theta$ to reach the second receiver antenna, as illustrated in Figure 9.5. This corresponds to a phase difference of $\Delta\phi = (2\pi/\lambda)\ell \sin(\theta)$ between the signals received at the two RX antennas. Given the measured phase difference $\Delta\phi$, the angle of arrival θ can be computed as

$$\theta = \arcsin \frac{\lambda \Delta\phi}{2\pi\ell}. \tag{9.10}$$

While one TX and two RX antennas are sufficient in principle for determining the angle to a target, having more than two RX antennas enables higher resolution and thus the ability to distinguish multiple nearby targets. The phase of the returned signal will be offset by an additional $\Delta\phi$ at each RX. Sampling the signal across the RX antennas and performing an FFT on this signal sequence can be used to reliably estimate $\Delta\phi$.

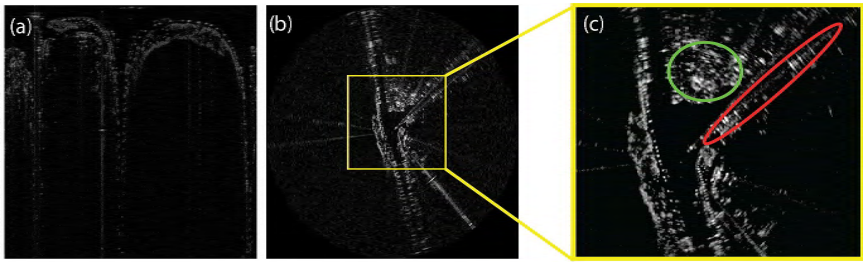

Figure 9.6 A polar radargram in (a) transformed to Euclidean coordinates in (b), where we can observe several types of radar noise that are unique compared to other sensors from a zoomed view in (c). Speckle noise returns are the most common, with ambiguous clutter circled in green. Multipath reflections develop where returns bounce off nearby walls or the ground before hitting the antenna, generating reflections of true targets. A series of repeated returns is circled in red. The original image is sampled from the Mulran dataset [575].

The above example illustrates a "SIMO" phased-array radar system (single input, multiple output). Most radar sensors used for SLAM applications are MIMO (multiple input, multiple output) with several TX and RX antennas. Rather than doubling the number of RX antennas, it is possible to achieve the same resolution by adding one more TX antenna, as long as the RX antennas can distinguish the signals from the multiple TX antennas. Different techniques can be used to ensure that the TX signals are uncorrelated (orthogonal); *e.g.,* frequency division (where each transmitter uses a different frequency band), code division (where each transmitter sends a signal modulated by a unique code sequence), or time-division multiple access (TDMA) where each transmitter uses a different time slot. The same principle can also be applied to 2D TX-RX arrays that can measure both azimuth and elevation angles, thus producing 3+1D data.

9.1.3 Challenges to Radar Applications

Radar technology, like any sensor system, presents a range of challenges that necessitate careful consideration during development. These challenges include a variety of noise types that are particularly prominent in radar, such as multi-path reflections, biased and sparse range readings due to the wide beam width, receiver saturation, and speckle noise. Illustrations of some of these noise phenomena are provided in Figure 9.6, which depicts a polar radargram transformed to Euclidean coordinates. We discuss some relevant radar filter techniques in Section 9.1.4.

9.1.3.1 Speckle Noise

Noise in radar measurements come from several sources, including thermal noise, electronic flaws, and varying RCS of targets. When a radar emits an electromagnetic

pulse, it captures the energy reflected back by all objects within the antenna's field of view. The interaction of this pulse with objects scatters the radar waves, leading to constructive and destructive interference. Such interactions can either produce false signals or cancel out legitimate returns received by the antenna, irrespective of the signal's origin. These factors contribute to signal variations across the frequency domain, where the most prominent peaks represent a mix of genuine targets and false alarms. In the absence of a mechanism to distinguish between genuine and false returns, the sensor ends up generating a pattern of scattered points, commonly referred to as speckle noise. For accurate identification of landmarks, crucial for pose estimation and feature matching, it becomes essential to estimate the uncertainty around these reflections, possibly over several scans.

9.1.3.2 Multipath

Beyond speckle noise, multipath returns constitute another form of measurement errors, originating from varied detection paths associated with a single object. Imagine a scenario with a landmark situated in front of the sensor. While some transmitted rays may directly reach this landmark, others might only arrive at the antenna after reflecting off the ground or bouncing off a wall. To the radar, it appears as though the landmark is located beneath the road or beyond the wall, leading to the perception of what are termed 'ghost objects' or static outliers. The elimination of these outliers is crucial for ensuring the reliability of point cloud mapping or localization.

9.1.3.3 Motion-Induced Distortion

Scanning sensors, including both lidar and radar, inherently exhibit motion distortion. This is particularly true for spinning radar, which constructs each polar image through a single rotation. Consequently, if the sensor is moving, the position of a single object captured at the start and end of one rotation will differ. This discrepancy becomes significant with sensors operating at low frequencies or when the vehicle moves swiftly. For instance, the Navtech CIR 304, a commonly used imaging radar, operates at a frequency of 4 Hz, posing challenges for accurately aligning raw frames. There are also faster spinning radars like the newer Navtech RAS3 which spins at 10 Hz, similar to many lidars, and the Indurad iSDR that can spin at up to 50 Hz. Still, when the sensor is mounted on a fast-moving platform (like a car), the motion-induced distortion can be significant.

9.1.4 Radar Filtering

The occurrence and distribution of false targets (Section 9.1.3) can change over time and are characterized by unpredictable parameters, rendering static filtering approaches such as simple thresholding insufficient, as they may allow false alarms to pass through. In response, researchers have devised methods to dynamically

estimate the distribution of false alarms, taking these challenges into account. These filtering methods typically consider the measurements along one azimuth direction, trying to estimate which range bin(s) contain true targets and which are false alarms.

The constant false alarm rate (CFAR) filter [817] is a popularly implemented method designed to sustain a specified probability of false alarms amidst dynamically changing and uneven interference. The process begins by segmenting a signal – such as the frequency domain representations obtained post-FFT of radar ADC samples – into discrete segments known as cells. These cells are then assessed using a sliding-window approach. At the core of this window lies the set of cells under test (CUT). The intensity of the CUT is evaluated against that of the adjacent cells, referred to as training cells, which precede and follow the CUT. In some implementations, guard cells may be placed between the training cells and the CUT to prevent the local influence of the CUT from affecting the training cells' magnitude. A decision to accept or reject a CUT set is made based on whether its intensity surpasses a calculated threshold, which is derived from the comparative intensity of the surrounding training cells, guard cells excluded. Several variants of CFAR exist, the canonical version being cell-averaging CFAR (CA-CFAR), where the threshold is computed in relation to the mean power of the training cells, scaled by a threshold multiplier that is selected from the desired probability of false alarm. A common alternative is ordered-statistic CFAR (OS-CFAR) where a more robust statistic such as the median of the training cells is used instead of the mean.

Figure 9.7 illustrates how the CA-CFAR threshold adapts to a radar signal and which range bins are selected as targets vs noise, compared to two other filtering methods.

A variant of CFAR designed and tested specifically for radar odometry is BFAR (bounded false alarm rate) [32] which simply modifies the output Z computed from a CFAR detector with an affine transformation $T = aZ + b$, where b is a learnable parameter that scales the output to blend between the CFAR output and a fixed-level threshold.

While CFAR and its variants are used in many radar applications, several pipelines for radar odometry and SLAM employ simpler filtering strategies. Whereas CFAR was developed for target detection (where it may be important not to miss a weak detection), when filtering radar for use in odometry the concern is rather to retain only those points that can reliably be detected over time and from different viewpoints. One popular technique involves selecting the k strongest returns above a static threshold along each azimuth, with k ranging from 1 and upwards. A statistical threshold may also be employed, selecting all points with an intensity higher than one standard deviation over the mean value.

Some recent approaches instead use machine learning techniques to increase the accuracy and resolution of radar output, typically using LiDAR data as ground truth for training. Cheng et al. [203] use a generative adversarial network (GAN)

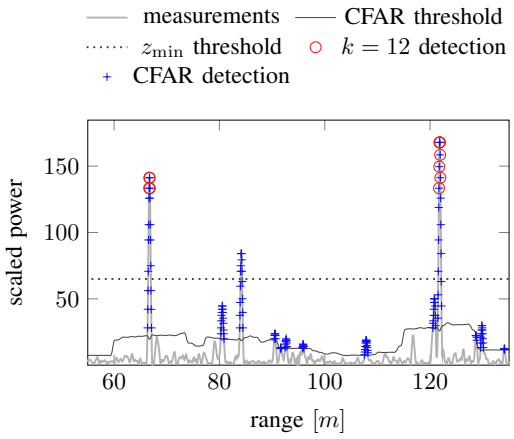

Figure 9.7 Examples of radar filtering, comparing a CFAR filter with a constant power threshold and a k-strongest strategy. The power/range plot along one azimuth direction is plotted in grey. Returns from true targets appear as spikes in the plot but there are also several ambiguous peaks. CFAR produces an adaptive threshold, plotted with a black line. Detections reported by the CFAR filter are plotted in blue, and in this case includes several "false alarms". The k strongest filter (with $k = 12$ in this case) is more conservative and only returns points around the main targets. Figure from Adolfsson et al. [13] (©2021 IEEE).

to generate point clouds based on range-Doppler velocity matrices. Xu et al. [1206] train a regressor and classifier, where the regressor outputs improved, higher-resolution depth readings, and the classifier provides an estimate of whether the data is out of range. These methods strive to learn models that can retain only those returns in the radargram that correspond to a surface that would be detected by a LiDAR.

9.2 Radar Odometry

The goal of radar odometry is to, given a set of ordered radar readings over time, estimate the egomotion of the sensor. A radar odometry approach typically involves handling an intermediate representation, such as a set of the last N radar readings or a continuously pruned local map. The focus lies on obtaining an accurate pose estimate at a local scale. Without considering explicit loop closures, the error will eventually accumulate without bounds even for good odometry methods. Methods using spinning radar may accumulate on the order of 1–2 % translational drift per 100 m.

A particular feature of many radar sensors is the per-point 'Doppler' velocity estimates, which can be used to estimate odometry in a correspondence-free manner, as described in Section 9.2.1. In addition to Doppler-based methods, the relative

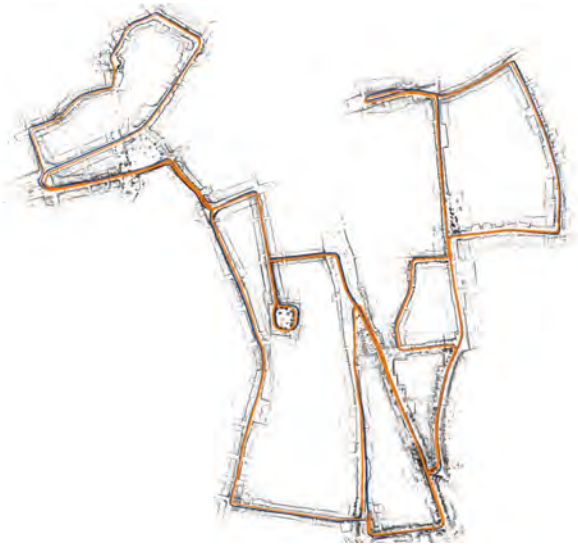

Figure 9.8 Example of open-loop radar odometry on the Oxford Radar Robotcar dataset [133], using the CFEAR method [15] with data from a Navtech 2D spinning radar. The ground-truth trajectory plotted in blue and the odometry estimate in orange. Point targets extracted from the radargram in grey.

transformation between two nearby radar scans is often estimated via spatial correspondences so as to determine which parts of one scan can be found in the other scan. Given a set of such correspondences, a distance metric can be computed and optimized. Depending on the type of scanner, how to obtain these correspondences is different; a spinning radar often produces a raw signal that either can be used directly as described in Section 9.2.2, to extract higher-level features containing information from the raw signal (see Section 9.2.3) or to extract range points that can be used in registration, much like in LiDAR odometry (see Section 9.2.4).

An indicative example of what open-loop radar odometry using a 2D scanning radar may look like is shown in Figure 9.8.

9.2.1 Doppler Odometry

The radial velocity obtained from Doppler measurements can be used to directly estimate the sensor's linear velocity. In Doer and Trommer [280], Doppler information is utilized via a combination of three-point RANSAC and a least squares problem to estimate linear velocities.

However, the rotational velocity component is not directly observable from the per-point velocity measurements in the data from a single radar – unless assumptions can be made about the kinematic model for the radar system, for example,

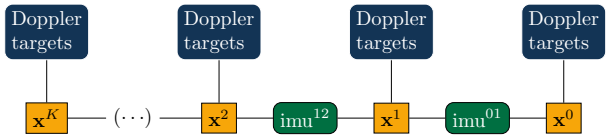

Figure 9.9 Factor graph representation of the radar-inertial velocity estimation system. States from K previous timesteps are jointly estimated using Doppler targets and sets of IMU measurements as constraints. Figure adapted from Kramer et al. [613].

knowing where the radar is mounted with respect to the center of rotation and assuming no skidding [559, 362]. Therefore it is common to use IMU data (or more specifically, a gyroscope) for radar odometry systems that relies solely on Doppler information from a single radar. In Huang et al. [497], a consumer grade IMU combined with cascaded SoC radars achieves low drift in diverse 3D indoor spaces. Kubelka et al. [617] compared several variants of registration-based approaches for 3+1D radar with registration-free Doppler + IMU odometry [280] and found the registration-free method to produce the lowest error, not least in feature-sparse environments, with a drift as low as 0.3% over a 4.5 km trajectory reported.

Kellner et al. [559] show how linear and rotational velocity can be estimated from 2+1D radar data when the radar is mounted on a vehicle with Ackermann steering and the mounting point of the radar sensor with respect to the center of rotation of the vehicle is known. Galeote-Luque et al. [362] extend this to the 3+1D case and estimate five degrees of freedom (linear motion in three dimension plus yaw and pitch rotation, but not roll).

Since the Doppler-based modality of odometry is rather specific to the radar methodology, we provide an example based on Kramer et al. [613].

Example: Doppler Odometry Factor Formulation

A straightforward approach to integrating a Doppler factor from radar is in estimating the body-frame velocity of the sensor platform over a sliding window of K previous radar measurements. These velocities are interconnected through integrated accelerometer measurements from the IMU, which can form a comprehensive system for accurate velocity estimation. The system's structure can be represented using a factor graph, where states from N previous time steps are jointly estimated using Doppler targets and sets of IMU measurements as constraints.

Accelerometer measurements are typically affected by both bias $\mathbf{b}_a$ and gravity $\mathbf{g}_W$. Since velocity estimates derived from radar data are free from bias, accelerometer biases can be compensated for by including them in the state vector. However, compensating for the effects of gravity requires estimating the IMU's attitude, specifically its pitch and roll, which are represented by the orientation quaternion $\mathbf{q}_{WS}$. To accurately estimate the IMU's attitude, gyro measurements are used, ne-

cessitating the estimation of gyro biases $\mathbf{b}_g$. Consequently, the full state vector is expressed as $\mathbf{x} = [\mathbf{v}_S^T, \mathbf{q}_{WS}^T, \mathbf{b}_g^T, \mathbf{b}_a^T]^T$.

The radar-inertial ego-velocity estimation is formulated as an optimization problem, where the cost function integrates the constraints and measurements to provide an accurate estimate of the sensor platform's velocity

$$J(\mathbf{x}) = \underbrace{\sum_{k=1}^{K} \sum_{d \in \mathcal{D}^k} e_d w_d}_{\text{Doppler term}} + \underbrace{\sum_{k=1}^{K-1} {\mathbf{e}_s^k}^T W_s^k \mathbf{e}_s^k}_{\text{inertial term}} \tag{9.11}$$

where K is the number of past radar measurements for which states are estimated, $\mathcal{D}^k$ is the set of targets returned from the radar measurement at time k, e_d is the Doppler velocity error, and $\mathbf{e}_s$ is the IMU error. The error terms are weighted by the information matrix W_s in the case of the IMU errors; and the normalized intensity of the corresponding radar target

$$w_d^j = \frac{i_j}{\sum_{d \in \mathcal{D}} i_d} \tag{9.12}$$

in the case of the Doppler velocity measurements where w_d^j is the weight for target j in scan $\mathcal{D}$ and i_j is the intensity of target j. In the following sections we detail the formulation of our Doppler and IMU measurement constraints.

In this example we consider radar measurements consisting of a set of targets $\mathcal{D}$. Each $d \in \mathcal{D}$ consists of $[r_S, v_R, \theta_S, \phi_S]^T$ representing the range, Doppler (radial) velocity, azimuth, and elevation for target d. The Doppler velocity measurement v_R is equal to the magnitude of the projection of the relative velocity vector between the target and sensor $\mathbf{v}_S$ onto the ray between sensor origin and the target $\mathbf{r}_S$. This is simply the dot product of the target's velocity in the sensor frame and the unit vector directed from the sensor to the target

$$\tilde{v}_R - e_v = \mathbf{v}_S \left(\frac{\tilde{\mathbf{r}}_S}{\|\tilde{\mathbf{r}}_S\|} \right)^T. \tag{9.13}$$

In this approach, it is assumed that the targets in the scene are stationary and only the sensor platform is moving. In this case each radar target can provide a constraint on our estimate of the sensor rig's velocity in the body-frame. The velocity error for each radar target is then:

$$e^k(\mathbf{x}^k, \mathbf{d}^{i,k}) = v_R^{i,k} - \mathbf{v}_S^k \left(\frac{\mathbf{r}_S^{i,k}}{\|\mathbf{r}_S^{i,k}\|} \right)^T \tag{9.14}$$

where x^k is the state at time k and $\mathbf{d}^{i,k}$ is the i^{th} target in the set of radar measurements at time k. As previously noted, radar measurements are affected by non-

Gaussian noise and radar scans often contain false target data. These challenges may be addressed by using the Cauchy robust norm with the Doppler residual.

9.2.2 Direct Odometry

Methods that operate on raw radargrams as the ones depicted in Figure 9.6 and Figure 9.2, as opposed to points filtered from the radargrams, are denoted as *direct*.

Direct approaches make use of classical signal processing techniques such as phase correlation and the Fourier-Mellin transform [182, 854]. Given two sequential polar radargrams, their relative rotation can be found by a translational shift in the polar coordinate frame – where a vertical shift corresponds to a change in azimuth angle. Using phase correlation, the relative orientation is selected from the pixel shift that maximizes the agreement of the two polar images. Subsequently, the translation can be refined by a similarly computing the correlation between the images in a Cartesian frame (Figure 9.6). These direct correlation-based methods assume that power returns from a specific location remain stationary over time, enabling meaningful correlation. However, this assumption often fails, especially with dynamic objects or in radar data, where noise artifacts are common.

Correlation is also used in the "Masking by Moving" method of Barnes et al. [60]. Two-dimensional correlation between the current scan and rotated copies of the previous scan is computed on a regular grid of pose candidates. However, Barnes et al. address the problem of nonstationary power returns by training a convolutional neural network (CNN) to avoid including false features that are due to noise. The CNN is trained to predict a mask that keeps only those parts of the radargram that are stationary and thus more useful for correlative scan matching.

In contrast to these direct odometry methods that use all or mostly all of the radargram, the methods in the remainder of this chapter are *indirect* in that they first select specific features or key points and operate on those sparser points to estimate the odometry.

9.2.3 Feature-based Odometry

Given that radargrams from 2D spinning FMCW radars are essentially birds-eye-view images, it is natural that several works utilize image-based feature extraction and matching techniques from the computer vision community to find correspondences and estimate odometry; such as SIFT [146, 657], SURF [481], and ORB [554] features. (Callmer et al. [146] match large-scale features of islands in an archipelago and Li et al. [657] extract features from a satellite radar.) Compared to camera images, the noise level is higher in radar data and highly dependent on the environment. Hence, extracting descriptors that can be used for feature matching is more difficult [482]. Feature descriptors may also be more place-variant in radar

compared to other sensors, making it difficult to do data association if the sensor pose is different. (See also Section 9.3.) FSCD and BASD [915, 988] are examples of key-point extractors and feature descriptors specifically designed for radargrams.

The techniques above involve extracting a set of salient key points (which can reliably be detected in subsequent frames) and computing a feature descriptor describing the surrounding region of the feature point. Given the extracted feature set, the correspondences are computed using feature descriptor matching, often combined with a robust estimator, such as RANSAC. Given the correspondences, the spatial distance between features is minimized, often by finding the least squares solution using Singular Value Decomposition (SVD). In general, a feature-based approach is more stable towards large initial errors compared to the registration based approaches discussed in the next section, since feature descriptors can be associated robustly compared to registration methods that primarily hinge on point proximity for data association.

Going beyond hand-crafted feature descriptors as in the examples above, key-point extraction and feature descriptors can also be generated via deep neural networks, thus allowing features to be automatically generated [59]. One drawback is that training data including radargrams along with ground truth poses from a somewhat similar environment are required.

9.2.4 Registration-based Odometry

There are well-developed methods for odometry estimation based on point clouds from LiDAR ranging sensors and similar techniques can be used for point clouds extracted from radargrams or radar datacubes. In this section we describe how such radar point clouds can be computed and used for odometry estimation.

For spinning radars that provide raw signal data as shown in Figure 9.6 we first need to select which points to use by filtering those signal returns that do not correspond to a relevant peak. (These filtering techniques can be seen as similar to the key-point extraction in Section 9.2.3 but without extracting descriptors.) As discussed in Section 9.1.4, many approaches extract a set of range readings per azimuth; *e.g.,* using CFAR (Figure 9.7, Section 9.1.4), using noise statistics to remove redundant or noisy readings [164], BFAR [32], or simply the k strongest returns per azimuth. An exception is Kellner et al. [558], using DBSCAN clustering so as to also consider neighboring azimuth angles instead of restricting the search for targets along one azimuth dimension at a time. Models trained with machine learning so as to estimate a point cloud similar to that from a LiDAR from a radargram are also commonly used in recent methods [203, 1206]. A key challenge is to extract an adequate amount of readings; too few readings discards relavant information and too many include noise [164]. For example, CFAR has been found difficult to tune in this respect [133].

Once a point cloud has been extracted using one of the techniques above, it

can either be used as-is or additional information can be estimated by examining the local surrounding region, such as normals, planes and point distributions. The registration approaches used for radar data are often similar to what is done using LiDAR based scan registration approaches (see Section 8.2); however, the noise level and sparsity in radar data makes pair-wise registration much more challenging.

A common strategy in registration-based odometry methods using radar data is to register new scans to multiple previous scans – either aggregated into a submap or as a set of individual point clouds. Registering to multiple scans is a way to compensate for several of the challenges in radar data as discussed above. By including more key frames, the odometry estimate is less sensitive to sparse and noisy radar point clouds. More correspondences adds more constraints which can reduce drift in feature-poor environments. Another goal is temporal redundancy, in the sense that sudden occlusions or spurious correspondences from moving objects impact the odometry estimate less when multiple key frames are used.

Some examples of registration-based radar odometry methods include continuous-time ICP [133], power-shifted NDT [623], and CFEAR [15] (not to be confused with the CFAR method for filtering which is discussed in Section 9.1.4) – all of which make use of submaps or multiple key frames. In CFEAR, point clouds are extracted from radargrams by selecting the k strongest returns along each azimuth. Each new cloud is registered jointly against the s most recent key frames using either a point-to-point, point-to-line, or point-to-distribution error metric – akin to NDT scan registration (see Part I). For each point, the normal vector is estimated from the covariance matrix of neighboring points within some radius. Point correspondences are weighted based on the agreement of their normal vectors, the planarity (condition number of the covariance matrix), and the number of points in the neighborhood. Kung et al. [623] use a fixed threshold to extract a point cloud from the radargram and aggregate multiple point clouds into a radar submap using an NDT representation where the contribution of each point is weighted by its returned signal strength. Burnett et al. [133] extract point clouds from radargrams using BFAR [32] and aggregate into a local submap, after which a continuous-time ICP formulation is used to optimize a trajectory estimate where points are associated with a Gaussian process motion prior.

The methods above are all based on 2D radar data. Recent pipelines for registration-based 3+1D radar odometry tend to adopt similar strategies – although point cloud extraction is performed on the sensor so the design choices for selecting which signal peaks to consider as valid targets come down to sensor-specific thresholds rather than explicit feature extraction. Since per-point Doppler speed information is available in 3+1D point clouds, these methods typically consider a least-squares estimate of the ego-velocity from Doppler data (see Section 9.2.1) as an initial estimate and perform point cloud registration to refine it. The registration-based odometry component in 4DRadarSLAM [1266] uses a variant of GICP [990] that is adapted for radar point clouds, where points are weighted by a covariance matrix that assigns

a higher uncertainty to points far from the sensor due to the limited azimuth and elevation angle accuracy. 4D iRIOM [1305] employ one-to-many distribution-to-distribution matching in order to alleviate the noise and sparseness of radar point clouds. Instead of matching each point to its closest corresponding distribution in the local submap, each point is matched to a weighted set of closest distributions. (The complete SLAM pipelines [1305, 1266] are further described in Section 9.4.2.) The EFEAR-4D method [1196] extends CFEAR 2D odometry [15] to 3+1D radar point clouds. After computing a Doppler-based ego-velocity estimate and removing outlier points that do not agree with this least-squares estimate and therefore can be assumed to come from moving obstacles, the remainder of the registration scheme is similar to CFEAR: registering scans to a sequence of preceding key frames and using agreement of normal vector and planarity for associating and weighting individual point matches.

9.2.5 Motion Compensation

As discussed in Section 9.1.3.3, it is important to compensate for motion distortion in odometry estimation. In the case of a low-speed spinning radar with a frequency of 4 Hz, egomotion compensation is reported to reduce ATE (Absolute Trajectory Error) by 29% by using a constant velocity model [15]. Given the time stamps of two subsequent radar scans and the relative pose computed using the methods above, a velocity can be computed and each radar point can be shifted accordingly, since the per-point timing is also available. Offsetting the points of individual radar scans in such a way compensates for the substantial motion that can be encountered during the slow sweep of a scanning radar. The same model is also used to provide an initial estimate to rigid scan registration, and after registering each motion compensated scan to the previous, it is added as a node in the SLAM pose graph.

However, this approach still works in a discrete pose graph setting. Continuous-time trajectory representations can be beneficial for obtaining a smooth and accurate trajectory where the pose estimate used for undistortion can be queried at the time of each sensory reading. In Ng et al. [809] a spline representation is implemented in a pipeline that uses automotive SoC radars. Gaussian processes are used in Burnett et al. [135] to form a factor graph representing the trajectory by combining an IMU sensor with a spinning radar.

9.3 Radar Place Recognition

As with other sensor modalities, place recognition (PR) is an essential module for radar SLAM. A good overview of the problem in general can be found in Chapter 8. As discussed in the chapters on visual and LiDAR PR, securing invariance to both translational and rotational change is crucial, such as when traversing a street in a different lane or arriving at a junction from another street.

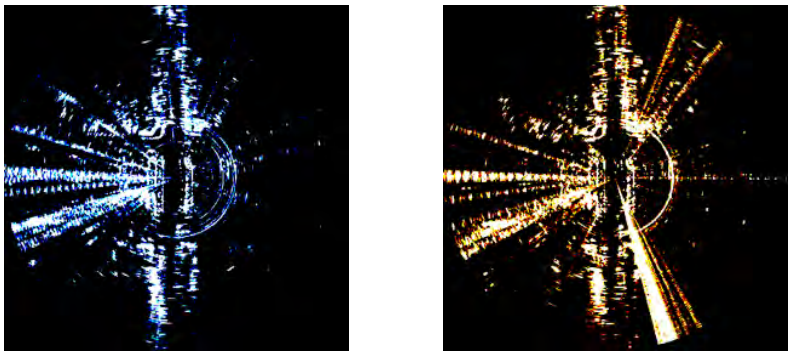

Figure 9.10 Two radar scans obtained from the same location. The noise pattern induces visual aliasing, complicating the process of place recognition.

9.3.1 Unique Challenges in Radar Place Recognition

While, in principle, several of the methods discussed in the earlier chapters could be applied to radar data after proper data format conversion, some unique characteristics and challenges exist (see also Section 9.1.3).

Several factors contribute to the key challenges in radar place recognition. Firstly, the low resolution of radar data results in less details and thus fewer distinguishable features for recognition. The wide beam of radar data contributes to angular ambiguity, causing distant objects to appear as broad patches that differ significantly when viewed from closer ranges. Additionally, a relatively low signal-to-noise ratio poses challenges for place recognition if adequate filtering is not applied, as demonstrated in Figure 9.6. Receiver saturation can produce a strong radial feature that varies significantly with the observation angle of a particular target, as illustrated in the sample image in Figure 9.10. Consequently, the appearance of a place can change notably from nearby positions due to these challenges.

As there are different types of radars, place recognition needs to be handled differently based on the type of sensor. How a place is perceived and described significantly differs between a long-range 360-degree spinning radar and a SoC radar with limited range and FOV. For example, a spinning radar produces a 2D radargram, which can be treated as an image. Naturally, approaches inspired by visual PR have been applied for the spinning radar. Applying a target detection algorithm such as CFAR to the radargram results in a point cloud, after which methods from 2D LiDAR PR can be adapted. Spinning radars tend to have a very long range which can be greatly beneficial for place recognition [575] yet their slow scanning rate may lead to motion distortion even at low driving speeds. SoC radars, on the other hand, have a faster update rate since they need not mechanically spin the antenna in order to cover their field of view, and are therefore less susceptible to motion distortion. Most methods that use SoC radar data for place recognition

work with a 2D/3D point cloud data format and not the full datacube. As a result, SoC radar PR often builds upon existing LiDAR PR methods; however, the measurements from SoC radars are typically restricted to a small FOV and tend to have higher sparsity and noise.

Currently there is more PR literature using spinning radars than SoC radars. This is in part because spinning radars capture richer information over a wider FOV, allowing them to better capture surrounding structures for robust PR, especially for outdoor applications, and in part due to the availability of large-scale datasets. However, this advantage does not prevent SoC radars from being used for PR. While spinning radars offer a larger FOV, their projection model only provides 2D information, losing elevation details. For indoor PR, where a shorter range and smaller FOV are sufficient, the 3+1D measurements from SoC radars can still be advantageous.

9.3.2 Learning-based Radar PR

For spinning radars, treating the 2D radargram as an image, 2D image retrieval has been leveraged for place recognition.[2]

Saftescu et al. [968] is an early approach to PR from 2D FMCW radar which uses a CNN to represent radargrams, specifically addressing their polar nature by using cylindrical convolutions in order to learn a representation that is rotation invariant. Then, a query radargram can be matched against a database of reference images to produce an appearance-only topological PR system. De Martini et al. [255] add pose refinement to produce a topometric mapping and localization system.

PR methods may be trained with augmented instances of the input data in order to be more robust to slight variations. Given that PR data typically is recorded sequentially, scan by scan while driving along a path, "augmented" instances can be retrieved by sampling frames that are sequentially nearby and artificially adding rotation by shifting the polar radargram. Contrasting to such augmented instances, the network is trained not only to recognize similar instances but also to distinguish instances (and their augmentations) that are sequentially far away. In this way, data for training a PR algorithm can be obtained in an unsupervised way, without knowing the true metric location of the radar data [360].

The methods mentioned above have been designed specifically for spinning radar providing 360-degree long-range coverage. SoC radars, which are more attractive for automotive applications given that they lack moving parts and have a smaller size, require slightly different treatment since they typically have lower range and smaller field of view.

Cai et al. [145] use a deep spatiotemporal encoder (after projecting the point cloud to a 2D image plane) to generate feature vectors as place descriptors for

[2] A more comprehensive overview can be found in [127].

topological PR. These vectors are passed through a NetVLAD [42] layer after being re-ranked based on the RCS to filter out non-relevant stationary features. In this case, multiple radar sensors are mounted around the vehicle to overcome the limited field of view.

Herraez et al. [462] demonstrate single-scan radar place recognition with a pipeline that addresses data sparsity and noise by learning to focus on salient points that are important for place recognition. Similar to Cai et al. [145], Herraez et al. [462] also leverage NetVLAD to generate a feature encoding. However, they capture 3D contextual information by using rigid kernel point convolutions, as opposed to projecting the 3D point cloud to a 2D image. Furthermore, a 'point importance estimator' outputs the probability of a point being important for place recognition. This estimator is trained with sets of known corresponding point clouds, and query points that have a correspondence within a small radius in the other scan are labelled important. RCS information is incorporated through an separate network that encodes RCS data from the points into a more compact feature representation. Peng et al. [865] filters noise points in a similar way as the Doppler odometry methods in Section 9.2.1. Using RANSAC to estimate the ego-velocity produces a set of inlier points which are likely to be stationary. The remaining outlier points can then be filtered as noise. Rather than using NetVLAD which was designed for visual place recognition, Peng et al. [865] demonstrate a radar-specific feature extraction backbone named MinkLoc4D which takes inspiration from LiDAR place recognition architectures [1315].

9.3.3 Descriptor-based Radar PR

Using hand-crafted descriptors instead of learned embeddings is often effective as well. A common approach for radar PR is to directly adopt LiDAR descriptors, such as Scan Context [576], RING [1208], or M2DP [453]; though proper adjustments are necessary due to the differing sensor data formats. For example, the 3D structural information (*e.g.,* height) is missing in spinning radar data, but can be replaced with RCS. Furthermore, additional care should be taken with radar sensor data to address its inherent challenges, often requiring careful noise filtering, sparsity handling and motion compensation.

Hong et al. [482] implement place recognition by adapting the M2DP descriptor [453] originally designed for 3D point clouds to 2D point clouds extracted from radargrams. Additionally, they investigate the distribution of points on the 2D plane to assess if a point cloud is likely to be distinctive or not. Performing PCA on the 2D points produces two eigenvalues that describe the spread of the points along each eigenvector. Point clouds where the eigenvalues are substantially different indicate cases where the scan lacks features in one direction (such as data from highway driving) and those scans are not considered for place recognition by Hong et al.

Jang et al. [518] modify the RING descriptor [1208] for radar. The descriptor generated by RING is in a sinogram form providing roto-translation invariance. The correlation between two sinograms should give the correct match for the PR problem; yet, the high level of noise in radar images may prohibit a naive comparison. Additional incorporation of auto-correlation has been shown to enhance the PR performance for radar images.

Adolfsson et al. [16] adapt Scan Context descriptors made from 2D radargrams in several ways. Firstly, they compute the sum of intensities for all points in a Scan Context bin as a way to encode both the intensity and the point density. Further, each descriptor is generated from an aggregated set of noise-filtered and motion-compensated polar images in order to mitigate some of the challenges listed in Section 9.3.1. Keeping only the k strongest returns along each azimuth direction provides a conservative filter that tends to remove a large part of the noise otherwise present in radar point clouds. Creating the Scan Context descriptor from multiple registered point clouds further addresses the sparse data remaining after this conservative filtering. De-skewing the point cloud via a constant acceleration model is important for generating comparable descriptors when using spinning radars while driving.

PR methods designed for 3+1D SoC radars also commonly implement a variant of the Scan Context descriptor [1266, 1305, 665]. However, as spurious radar points can easily distort the height measurements, using the maximum height per radial bin (as in the original Scan Context) is not always as effective for radar point clouds as for LiDAR. The Intensity Scan Context descriptor [1151] is an alternative that stores the maximum measured intensity value of the points in a Scan Context bin rather than the height. Alternatively, the sum of intensities within a bin can be used, so as to use both the intensity and the point density, thus being able to encode vertical structures in the descriptor without being as susceptible to noisy point positions. Another important factor when used with 3+1D radar is that the modified radar descriptor should cope with a much narrower FOV compared to the 360° LiDAR that Scan Context was designed for. Given that a place will appear quite differently when observed from two different viewpoints, it is difficult to achieve rotation invariance when the sensor only covers a small FOV. One way to address this is to apply loop pre-filtering based on the current odometry estimate, only attempting to match the current descriptor to those of frames within a certain range of yaw angles (*e.g.,* 20°) [1266].

9.4 Radar SLAM

Radar SLAM systems generally implement the same overall structures as LiDAR- or camera-based SLAM solutions. In this section we briefly describe notable systems from the past two decades with a focus on aspects that are particular to tailoring

SLAM to radar data. We also describe multi-modal systems that combine radar with other exteroceptive sensors in Section 9.4.3.

9.4.1 Map Representations

Many radar SLAM systems generate maps that rely on similar representations discussed in Chapter 5. However, the sparse and often spurious nature of radar measurements introduces unique challenges in the mapping process. Some earlier works in radar mapping establish a map representation directly from existing target detection models, often using landmark maps where individual detected targets constitute the map [232, 279, 146, 988]. This target detection can also be used for occupancy grid map as in the series of works by Mullane et al. [788], while exploiting the detection probability [789] in the mapping phase.

A detected target, represented by an individual peak in the signal, can also be treated the same way as a point from a LiDAR scan. For instance, these points can be used to create 2D occupancy grid maps [737, 768] or point cloud maps [482]. The extracted point cloud can also be augmented with additional information, as in [16] which also computes the distribution of surrounding points to estimate orientation and weights, similar to surfels or NDT cells. Direct point cloud representations are also popular in emerging high-resolution SoC radars, which feature a larger vertical field of view (3+1D) and a larger number of TX/RX antennas (*e.g.,* 48 TX + 48 RX antennas for the Sensrad Hugin radar) [1266, 1305].

While less common, the full radar heatmap – comprising intensity samples for each direction and range bin prior to peak detection – can also be used to provide a dense grid map [955]. In this case, the map represents a global heatmap of the reflected power at each point in the environment, rather than evidence of occupancy. The 2D alignment between these heatmaps is then achieved through correlation.

Kramer and Heckman [612], in addition to generating odometry, presented a novel sensor model for voxel based mapping of radar data, capable of creating sparse maps even through visual occlusions. The sensor model leverages the log-odds based estimation of occupied vs free cells used in Octomap [486], but replaces their ray-cast model. The Octomap ray-cast model assumes that the first contact of a sensor is the only relevant point of occupation, but the generalized model accounts for radar's ability to penetrate certain material types by updating voxel probabilities within the sensors field of view, increasing probabilities in cells with radar returns and decreasing probability with missed scans, without assuming information along a ray. A related grid-map representation specifically designed with radar in mind is due to Nuss et al. [821] who designed a state estimation filter to address dynamic obstacles in grid maps called a probability-hypothesis-density multi-instance Bernoulli filter. This filter casts grid cells as a finite stochastic set, and fuses radar and lidar data dynamically.

Lastly, the challenges of lower density and higher noise in radar data can be

addressed using machine learning techniques [1206, 779], where LiDAR data serves as ground truth to achieve higher resolution and reduced noise in radar outputs. Mopidevi et al. [779] build a global radar map, where patches of the map are upscaled using a predictive network that filters noise from free-space regions and fills in sparse and empty regions, generating a map more similar to what can be obtained with LiDAR data.

Recently, neural fields, originally developed for RGB data [771] and later applied for LiDAR data [1288, 1061] have been applied to 2D radar data as well [104]. The key feature of neural representations is that they implicitly represent the environment such that the neural network can be queried with a point in space and return a quantity such as the distance to the closest surface, the colour and opacity, etc. In the radar fields of Borts et al. [104], a physics-informed radar sensor model as used to create an implicit neural geometry and reflectance model which can then be used to synthesize radar measurements from unseen view points. The received power at the radar sensor depends on the known transmit power and antenna gain but also the RCS which is composed from the size, radar reflectivity and directivity of the object. The neural representation learns to decompose the measured RCS into size (area) on the one hand and the product of reflectivity and directivity on the other hand.

9.4.2 Radar SLAM

In the preceding sections we have covered the main components that make up graph-based radar SLAM frameworks: open-loop odometry estimation, place recognition for loop closure detection, and map representations—in addition to some of the physical and technical principles that are pertinent to radar SLAM.

This section reviews a number of complete SLAM pipelines that use radar as the only exteroceptive sensor and discusses how they implement the components. These pipelines generally follow the same architecture as shown in Figure 9.1, with a front-end that has a filtering process to the raw radar data into a point cloud and a place recognition module to identify loop closures and add the corresponding constraints to the underlying pose graph, and a back-end mapping module that performs frame-by-frame odometry and SLAM-proper module that globally optimizes the map when loops have been detected.

Working with 2D radargrams from a spinning radar, the TBV-SLAM ("trust but verify") pipeline [16] builds upon the CFEAR 2D radar odometry method discussed in Section 9.2. The pose of the sensor is tracked using every radar scan in sequence but in the interest of efficiency, a sparser set of key frames is included in the SLAM pose graph. Once the estimated traveled distance exceeds a certain threshold (*e.g.,* 1.5 m) a new key frame is added to the pose graph and an odometry constraint is created based on the alignment to the latest key frame. As usual, a constraint in the graph requires both the relative pose offset between the two nodes and the

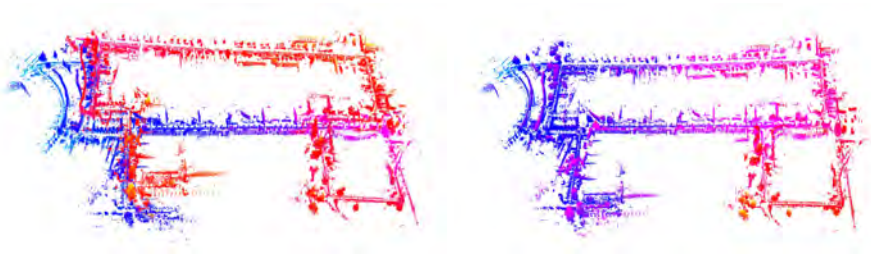

Figure 9.11 Example 3D map produced with radar SLAM using 3+1D radar (Sensrad Hugin) and IMU input. Color denotes height. *Left:* before loop closure. Note the accumulated horizontal and vertical drift that is evident in in the left part of 3D map. *Right:* after loop closure.

associated uncertainty expressed as a covariance matrix. Interestingly, it has been shown [16] that using a predefined diagonal covariance matrix with small values performs better than estimating the covariance based on the Hessian of the registration cost function, which might otherwise be expected to better capture the uncertainty stemming from the shape of the input point clouds such that a pair of point clouds from a tunnel would give a larger uncertainty (along the tunnel's direction), for example. However, using the Hessian tends to under- or overestimate the uncertainty which may cause the back-end optimization to slightly misalign the key frames. A central part of the TBV-SLAM pipeline is the place recognition module, where several candidate loop closures are retrieved ("trusted") and later tested after which the verifiably best candidate is selected. The Scan Context descriptor [571] (see 9.3.3) is adapted to account for both point density and signal strength (in lieu of the height data that is unavailable in 2D radar). Additionally, several techniques are implemented for retrieving and verifying loop candidates. For each key frame, several augmented descriptors are created by shifting the point cloud by lateral translation offsets. When searching for loop closures, the query descriptor is matched to all the augmented descriptors currently in the database, in order to account for loop closures where the vehicle is driving in a different lane. Loop candidates found by matching descriptors in this way are then filtered based on the odometry estimate (which hinges on having accurate enough odometry over large distances). After propagating the uncertainty from the odometry constraints between the query and candidate, candidate loop closures where the two scans are estimated to be far apart can be discarded. However, *jointly* considering the descriptor similarity and odometry uncertainty further improves the robustness of loop retrievals. That is, instead of finding the most similar candidate c to a query descriptor q such that $c = \arg\min_c d_{\text{descriptor}}(q, c)$ and then filtering based on odometry, the candidate is found from $c = \arg\min_c d_{\text{descriptor}}(q, c) + d_{\text{odometry}}(q, c)$, where d_{odometry} is computed as the likelihood of frame q being at the same place

as c taking the accumulated odometry uncertainty into account. Pairs of radar scans thus found are then aligned with the CFEAR registration module and finally, an alignment *verification* module which includes overlap measures as well as the CorAl [14] measure trained to detect slight misalignments. All in all, this pipeline demonstrates a number of techniques used to adapt descriptor-based place recognition to radar data (taking into account the multiple signal returns available in a 2D radargram and compensating for the comparatively sparse and slow scanning) and to sift through multiple loop candidates in order to verify the best ones based on geometric alignment as well as the front-end pose estimate.

The RadarSLAM pipeline of Hong et al. [482] is another prominent example of radar SLAM with 2D spinning radar data. In this pipeline, odometry (open loop pose tracking) is achieved by tracking key points detected directly in the radargram. A blob detector generates key points which are then tracked from frame to frame with a Lucas–Kanade tracker [715]. From the traveled distance computed by the tracker, a constant velocity model is used to compensate for the motion during one revolution of the scanner, and transformed key points are stored in the factor graph together with the poses of the key frames. Key frames are, as above, selected based on traveled distance. While pose tracking is done with a sparse set of key points, for loop closure detection denser point clouds are extracted from the radargrams. This point cloud extraction is done similarly as in TBV-SLAM [16]; however, instead of selecting the k strongest points per azimuth, RadarSLAM selects all points with an intensity higher than one standard deviation over the mean value. M2DP descriptors [453] are then created from the point clouds of the key frames, and loop closures are detected by matching frames with similar descriptors. As a safeguard against matching nondescriptive point clouds, RadarSLAM avoids selecting loop closures from key frames that are too elongated; *i.e.,* where the two eigenvalues computed from PCA are markedly different, since such point clouds are expected to be from non-unique places like a highway section. No other loop verification is performed.

One example of a SLAM pipeline based on SoC 2D radar is due to Schuster et al. [988]. Differently to the methods above, they maintain a graph that consists not only of the sensor poses but also individual radar feature nodes, whereas TBV-SLAM and RadarSLAM only optimize a graph of poses (although radar points are associated to the pose nodes in order to facilitate place recognition and rendering of the map). As 2D SoC radars generally provide far fewer detections than 360-degree spinning radars or 3+1D SoC radars, maintaining all detections in the optimizable graph is more feasible here. Edges are added in the graph to represent observations between all concurrently observed features. Their landmarks are extracted using a binary annular statistics descriptor (BASD [915]). As BASD is a compact binary descriptor, it is feasible to directly compare the descriptors of all feature points in a local region so as to associate recent features with those already in the map. Those features that pass a RANSAC outlier rejection stage are added as vertices to the graph, along with a pose node with odometry information from wheel encoders.

Assuming moderate drift from open-loop tracking, no explicit place recognition step is included, but point features can be matched with the BASD descriptors after loop closure, after which the SLAM graph is optimized in the back-end.

Two recent approached to 6DOF radar SLAM with 3+1D SoC radar are 4DRadarSLAM [1266] and 4D iRIOM [1305], both using 3D radar point clouds as input, where the point detection (filtering of the datacube) is handled onboard the sensor itself, so CFAR or other peak detection is not explicitly included in the SLAM pipeline. Still, the input point cloud may contain a lot of noise points. The Doppler radial velocity information included in the 3+1D point clouds is exploited by both methods to filter points from moving objects. The vehicle's ego-velocity can be estimated from linear least squares of the measured Doppler point velocities, and outlier points for which the velocity model does not agree are removed. iRIOM further denoises the point clouds by keeping as inlier points only those that have sufficiently many neighbor points (within a fixed radius) and where those points are compactly distributed (considering the covariance matrix of their spatial distribution). The Doppler ego-velocity estimation is used as a prior to a scan-to-submap registration step. 4DRadarSLAM uses a variant of GICP [990] termed APDGICP where points are weighted by a covariance matrix that assigns a higher uncertainty to points far from the sensor due to the limited azimuth and elevation angle accuracy. The submaps (key frames) are inserted into a graph. Loop closures are detected by Scan Context matching, and as opposed to the 2D methods above, the original Scan Context descriptor that includes point elevation can be used. While 4D iRIOM uses the original Scan Context, 4DRadarSLAM uses Intensity Scan Context [1151] in order to avoid uninformative descriptors due to noisy elevation measurements. 4DRadarSLAM additionally includes a validation step to reject candidate matches returned by Scan Context if the accumulated odometry distance between the two frames is above a certain threshold.

9.4.3 Multi-modality in Radar SLAM

So far we have mostly been concerned with methods for radar SLAM that use only radar data and in some cases propriocetive sensing like IMU or wheel odometry. In this section we discuss systems that combine mmWave radar with other exteroceptive sensors (*e.g.,* LiDAR, camera) or external data (satellite imagery or prior maps).

Some works in radar mapping have employed radar–LiDAR fusion so as to generate the best possible set of points given the current visibility conditions (trusting LiDAR more in clear conditions and radar more in low visibility). Fritsche et al. [350, 351] fuse the sensors based on estimated ranges to determine which sensor to trust in a given case. Radar and LiDAR have also been combined to improve place recognition, overcoming different sensor modalities by registering radar to LiDAR maps [1242, 1243].

Doer and Trommer [281] extend ROVIO [96], which is a filter-based visual-inertial odometry approach, to integrate radar egovelociy estimates using the Doppler odometry approach described in Section 9.2.1 [280]. In a similar way, also thermal camera data can be fused with radar to achieve a multi-modal radar-thermal estimation pipeline [281]. Zhang et al. [1267] combine data from thermal camera and a 3+1D radar point cloud in order to get robust frame-to-frame odometry in low-visibility settings. A transformer-based feature matcher detects corresponding points in sequential thermal frames and the radar point cloud is used to improve the depth estimate.

In terms of using multi-modal data for radar-based navigation, it is also worth mentioning methods that make use of overhead images and road maps, although most works in the literature exploit this kind of data specifically for localization, rather than for full SLAM. Hong et al. [483] demonstrate how 2D scanning radar data can be used to localize in prior public maps such as OpenStreetMap. Their system runs odometry using RadarSLAM [482] and represents the estimated pose of the sensor as a Gaussian mixture model. Line segments corresponding to building walls are extracted from OpenStreetMap which is used as a prior. Oriented points that are extracted from the radargram are then matched to the features of the prior map. However, given that the prior map information is uncertain and may be incomplete or outdated, this point-to-feature data association is challenging. Poses are sampled from the Gaussian mixture model of the current pose estimate and for each pose the oriented points of the current radar scan are matched to the line features of the prior, which localizes the scan to the prior. Another method that uses prior map data, in this case satellite imagery, for localising 2D radargrams is RSL-Net [1073], which consist of a set of deep neural networks. The first generates a synthetic image from an input overhead photo, showing what a radargram from that place might look like. Another networks estimates the relative rotation between a real radargram and an overhead photo (via the synthetic radargram generated in the previous network). Finally, another network estimates the relative translation offset between the radargram and the overhead image.

Some systems make use of so-called ultra-wideband (UWB) radio sensing. While UWB also uses electromagnetic waves within the radio spectrum and is sometimes referred to as UWB radar, the ranging capabilites of UWB differ dramatically from mmWave radar. When used in a radar SLAM framework, UWB radar is mostly used to detect similarities between sensor readings from different places. The frequency reponse after sending out a wide-band and wide-beam signal can provide a signature of the current location. The metric information is instead derived from wheel encoders or IMU data. Schouten and Steckel [984] and Takeuchi et al. [1070] use a database of UWB wave signatures to detect revisited places, For each place (node), a signature from the radar echo is stored along with the estimated pose. A graph is then created and optimized with odometry and loop constraints. Both approaches rely on odometry to obtain distances between nodes (*i.e.,* edges in a

graph) for metric SLAM and the signatures are created using a pulse-echo UWB sensor. Premachandra et al. [893] conversely use UWB radar to detect point features to be used in a landmark-based SLAM framework. They make use of multiple radar modules on each side of the robot and use trilateration of matched peaks in the signals from the sensors on either side to detect landmarks. In addition to the above, several UWB-based localization approaches use anchor-tag sensor configurations, where anchors are fixed to known locations and a battery-powered UWB tag is mounted on the robot or the asset to be localized. However, as this approach requires preinstalled infrastructure it is not directly related to SLAM.

9.5 Radar Datasets

In this section, we briefly summarize notable datasets from the radar SLAM literature. The datasets as also listed in Table 9.1.

Spinning Radar The datasets with spinning radar in Table 9.1 all use Navtech 2D radar sensors. Two of the first large-scale radar datasets for odometry and SLAM are the Oxford Radar Robotcar dataset [61] and MulRan [575]. These datasets have both been quite well used in the literature. The Oxford dataset covers a set of traversals of an urban driving route, totaling 280 km in various weather conditions. The MulRan dataset covers a more diverse set of environments, both dense urban and more rural driving, and longer time spans between sessions, but less driving in total. MulRan focuses on facilitating PR research but has also been well used to benchmark odometry and SLAM methods. The Boreas dataset [134] includes data from driving a route repeatedly over the course of one year (385 km in total), notably including adverse weather conditions such as snow and rain. In addition to SLAM-related benchmarks, this dataset also includes benchmarks for object detection (cars, pedestrians, cyclists). The Oxford Offroad Radar Dataset [361] is focusing on non-urban driving, in contrast to the other datasets in Table 9.1. This dataset covers about 154 km driving on unpaved roads and mountain trails in unpopulated areas.

SoC Radar Several SoC radar datasets are also available, both with 2+1D and 3+1D data. Some datasets geared towards autonomous driving focus primarily on object detection but have also been used for developing and testing SLAM approaches. NuScenes [143] combines data from five Continental ARS408-21 radars mounted on the car used for data collection with one LiDAR and six cameras. The dataset focuses on urban driving, in four cities, and notably includes annotated labels for object detection of 23 object classes. RadarScenes [987] is a dataset with four 77 GHz automotive 2+1D radars (unnamed) and one camera. It focuses on semantic perception and contains labels for 11 object types, but lacks accurate ground truth as well as IMU and LiDAR data. ColoRadar [614] is a radar SLAM dataset

	Dataset	Lidar	Cameras	Ground truth	Environment	Inclement Weather
Spinning radar	Oxford Radar RobotCar [61]	Yes	Stereo/Mono	GPS/IMU + VO	Dense Urban	🌧, 🌫
	Boreas [134]	Yes	Mono	GPS/IMU + RTK	Sparse Urban	🌧, ❄, 🌫
	MulRan [575]	Yes	No	SLAM	Mixed Urban	–
	RADIATE [1007]	Yes	Stereo	GPS/IMU	Mixed Urban	🌧, ❄
	OORD [361]	Yes	Mono	GPS	Urban and Offroad	❄, ●
SoC array radar	nuScenes [143]	Yes	Stereo	GPS/IMU	Mixed Urban and Natural	🌧
	RadarScenes [987]	No	Mono	None	Mixed Urban Roadways	🌧, 🌫
	ColoRadar [614]	Yes	No	SLAM	Varying	–
	NTU4DRadLM [1268]	Yes	Mono	SLAM	Mixed Urban	–
	MSC-RAD4R [220]	Yes	Stereo	GPS + RTK	Mixed Urban	🔥, ❄, ●
	Snail [490]	Yes	Stereo	TLS	Roadways and Tunnels	🌧, ●
	K-Radar [842]	Yes	Stereo	GPS/IMU + RTK	Roadways	🌧, ❄, 🌫, ●
	TruckScene [332]	Yes	Stereo	GPS/IMU + RTK	Roadways	🌧, ❄, 🌫, ●
Both	HeRCULES [577]	Yes	Stereo	GPS/IMU + RTK	Mixed Urban and Natural	🌧, ❄, ●

(🌧: Rain ❄: Snow ●: Night 🌫: Fog 🔥: Smoke)

Table 9.1 *Overview of public radar-related datasets. In the 'ground truth' column, VO denotes visual odometry, TLS denotes survey-grade terrestrial laser scans, RTK indicates GPS with real-time kinematic corrections.*

with data from a 3+1D Texas Instruments MMWCAS-RF-EVM board as well as 2+1D Texas Instruments module, in addition to IMU and 3D LiDAR data. Notably, this dataset includes raw analog-to-digital converter (ADC) values from the radar sensors in addition to 3D 'heat maps' (data cubes) and individual point targets. It covers both indoor and outdoor data, as well as data from an underground mine, and includes 6-Degree of Freedom (DoF) ground-truth tracking for pose estimation. NTU4DRadLM [1268] and MSC-RAD4R [220] both include high-resolution 3+1D radar data from an Oculii Eagle sensor. NTU4DRadLM covers structured (university campus) and unstructured (park) environments and MSC-RAD4R covers urban and rural on-road driving. The Snail-Radar dataset [490] features two high-resolution 3+1D radars: both Oculii Eagle and Continental ARS548, and includes data from handheld collection and on-road driving in urban environments.

9.6 Outlook and challenges

Radar SLAM pipelines that work in 3D are still rather few but as high-resolution SoC sensors develop we can expect there to be more work on fully 3D odometry and place recognition for radar. This will be particularly important on drones, for example, where radar is less utilized today. Creative designs will be required to enable 3D wide field-of-view radar units. In the meantime, we are likely to see significant advancements in handling constellations of several small field-of-view (FoV) radars in SLAM. This naturally comes along with calibration and other issues.

There will also likely be a shift towards making better use of the raw radargrams and datacubes, which contains spectral data. Traditional radar SLAM systems often resort to generating point clouds to interpret the environment. However, future systems are expected to take fuller advantage of the rich spectral information available

in radar data, providing more detailed and nuanced maps and improving both object detection and classification. One challenge to this is convincing manufacturers to open up access to the raw output of their products for research.

Radar semantic segmentation may also play a more prominent role in place recognition. By leveraging segmentation techniques, SLAM systems can more effectively differentiate between various types of objects in the environment, allowing for more intelligent navigation and decision-making. This will also help in reducing ambiguities in radar returns, leading to more reliable mapping.

Finally, there will be a greater focus on multi-modal data approaches, which integrate radar data with other sources of information including other sensors and also geographic priors (*e.g.,* OpenStreetMap). For example, by combining radar observations with these priors, radar SLAM systems may be able to localize more accurately in large-scale outdoor environments, further enhancing the robustness and reliability of autonomous systems.