# ORACLE: Occlusion-Resilient and Self-Calibrating mmWave Radar Network for People Tracking

Marco Canil, *Graduate Student Member, IEEE*, Jacopo Pegoraro, *Member, IEEE*,
Anish Shastri, *Graduate Student Member, IEEE*, Paolo Casari, *Senior Member, IEEE*,
and Michele Rossi, *Senior Member, IEEE*

***Abstract*—Millimeter-wave (mmWave) radar sensors are emerging as valid alternatives to cameras for the pervasive contactless monitoring of people in indoor spaces. However, commercial mmWave radars feature a limited range (up to 6–8 m) and are subject to occlusion, which may constitute a significant drawback in large, crowded rooms characterized by a challenging multipath environment. Thus, covering large indoor spaces requires multiple radars with known relative position and orientation and algorithms to combine their outputs. In this work, we present ORACLE, an autonomous system that: 1) integrates automatic relative position and orientation estimation from multiple radar devices by exploiting the trajectories of people moving freely in the radars' common fields of view and 2) fuses the tracking information from multiple radars to obtain a unified tracking among all sensors. Our implementation and experimental evaluation of ORACLE results in median errors of 0.12 m and 0.03° for radar location and orientation estimates, respectively. Fused tracking improves the mean target tracking accuracy by 27% and the mean tracking error is 23 cm in the most challenging case of three moving targets. Finally, ORACLE does not show significant performance reduction when the fusion rate is reduced to up to 1/5 of the frame rate of the single radar sensors, thus being amenable to a lightweight implementation on a resource-constrained fusion center (FC).**

***Index Terms*—Indoor sensing, millimeter-wave (mmWave) radar network, people tracking, radar fusion, self-calibration, sensor fusion.**
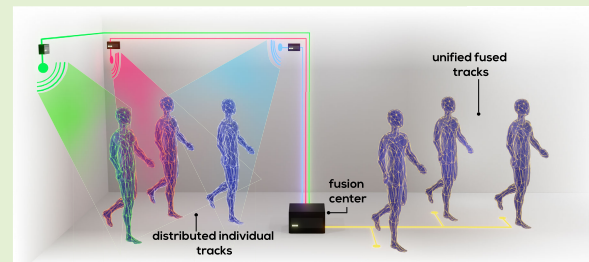
I. INTRODUCTION

RADARS operating in the millimeter-wave (mmWave) frequency band have emerged as valid alternatives to cameras for indoor monitoring, as they are robust to changing and poor lighting conditions, and raise less privacy concerns [1], [2], [3]. Their use enables advanced sensing applications spanning contactless people tracking [4], personnel recognition [4], [5], [6], and movement classification [7]. However, commercial mmWave radars have limited range [5] (up to 6–8 m) and are subject to occlusion [4], which may constitute a significant drawback in large, crowded rooms containing furniture and walls. Thus, covering large indoor spaces requires multiple radars (i.e., radar networks), with *known* relative position and orientation and algorithms to combine their output information.

In this work, we tackle the largely unexplored design of distributed mmWave radar networks to monitor people in indoor spaces. We aim to develop automatic calibration and sensor fusion algorithms to enable the quick deployment of multiple, jointly operating radars with *no human intervention* and *no accurate synchronization* between the devices. Multistatic radars used in, for example, [8], and [9], require synchronizing the devices' clocks to allow *coherent* processing of the received signals. This is highly impractical, as it mandates clock distribution through an optical connection to be set up, jeopardizing the ease and speed of deployment.

For this reason, we rather assume that the radar devices operate independently (i.e., they can only receive *their own* transmitted signals) and communicate the result of the target detection and tracking steps to a fusion center (FC). In this scenario, we need to solve two main issues: 1) automatically obtain the positions and orientations of the radars (*self-calibration*), as they are often unknown, or it is impractical to measure them at deployment time and 2) combine the environment perception capabilities of the multiple radars (*sensor fusion*), to boost their sensing accuracy and mitigate occlusion. The few existing solutions to point 1) present significant practicality and usability limitations in real scenarios [10], [11], [12]. Point 2), instead, has not been investigated with indoor mmWave radars, to the best of our knowledge. This aspect is particularly challenging as we aim at enhancing the tracking accuracy *without leveraging coherent processing*, thus only assuming coarse synchronization as provided by popular network protocols (e.g., the network time protocol, NTP).

In this article, we propose ORACLE, a solution to the mmWave radar network deployment and integration problem. Our contribution is twofold. As a first step, ORACLE automatically estimates the location and orientation of multiple radars with respect to a common coordinate system through an improved version of our previous work [13]. For this, ORACLE takes the trajectories of people moving in the environment as a reference. Then, the system fuses the information about moving people tracked by different radars at an FC, enhancing the resilience of the tracking process in the case of occlusion. ORACLE processes local information, transmitted by the radars to the FC, in a slotted-time fashion, thus handling the high variability in the frame rate of commercial radars. Then, it merges local tracks and provides a global representation of the moving targets in the environment.

The original contributions of this work are as follows.
1) We propose ORACLE, a novel plug-and-play system for the real-time, automatic self-calibration and integration of multiple *incoherent* mmWave radars for indoor people tracking.
2) As a first component of ORACLE, we present a fully automated method for the self-calibration of multiple mmWave radars. The algorithm extends our previous work [13] by adding a *masking* phase (see Section IV-C3) that handles a wider range of cases and provides better calibration results. ORACLE estimates the relative positions and orientations of the radars with a median error of 0.12 m and 0.03°, respectively, when three people move in the environment.
3) ORACLE includes a track-to-track radar fusion algorithm that combines information about the same subject collected by different radars. This improves the mean tracking accuracy by up to 27% with respect to single-sensor tracking.
4) We evaluate our method via an extensive measurement campaign through the RadNet platform [14], using four commercial mmWave radars deployed in realistic conditions and multiple subjects, including challenging human motion. In the most difficult case of three subjects moving concurrently, ORACLE achieves a tracking accuracy of 87% and a mean tracking error of 23 cm.

The remainder of this article is organized as follows: Section II provides a summary of the related work. In Section III, the problem tackled by ORACLE is presented and formulated. Section IV presents and discusses ORACLE, the proposed method. Section V provides some insights regarding the practical implementation of ORACLE and presents the experimental results on our testbed. Finally, conclusion is drawn in Section VI.

II. RELATED WORK

A. Multistatic Radars

Using multiple radar receivers with synchronized clocks enables coherent analysis of the received signals, which yields significant processing gains due to spatial diversity [15]. Existing works have leveraged this principle for drone detection [16] and people tracking [8], [17], [18], [19]. Despite their potentially superior accuracy and resolution, the main drawback of multistatic radars lies in their impracticality and deployment cost. Indeed, a common clock source needs to be distributed to the receivers, either via optical links or GPS, which is not available indoors. This would prevent the radar sensors from being quickly deployed, used, or relocated. Conversely, we target a scenario where ease of deployment and minimal human intervention are key requirements. For this reason, multistatic radars are not applicable and ORACLE focuses on track-level sensor fusion from incoherent sensors that are only coarsely synchronized (i.e., not at the clock level) using standard NTP.

B. Radar Networks

A large body of work has considered the use of incoherent radar networks outdoors, for example, in airborne and automotive applications, to improve the detection and tracking capabilities of the standalone sensors [20], [21], [22], [23]. These works tackle the fusion of distributed radar tracks without leveraging the multistatic gain available with precise synchronization. However, the considered radar setups are significantly different, as mmWave radars are typically used indoors or in short-range (6–8 m) outdoor scenarios, where the presence of multiple subjects may create crowded scenes and occlusions. The latter occurs when people or objects block the line of sight between a radar and the target, thus making the occluded subject undetectable. In outdoor scenarios, occlusion seldom occurs as targets are typically widely separated, the monitored area is much larger, or a lower frequency band is used for better propagation at longer distances (see, e.g., [20], and [21]). On the other hand, most existing works on mmWave radar networks rely on very simple offline data association rules based on *known* sensor positions, with no data fusion to improve the tracking accuracy [24], [25]. This is reasonable in large-scale outdoor scenarios or automotive applications, as the radar sensors are supposed to be deployed in a fixed location *for good*. Conversely, practical human sensing applications require a more flexible system where sensors can be moved around frequently and can automatically

recalibrate for the new deployment. To the best of our knowledge, only two works have addressed indoor radar networks for people tracking [12], [26]. A major drawback of both is that they consider *only one person* to be present in the environment, thus neglecting the reciprocal occlusions caused by crossing moving trajectories of multiple subjects. This is unrealistic in general indoor settings. All the above limitations are solved by the proposed solution, which is the first system that: 1) combines the information from *multiple radars*, handling the presence of *multiple subjects*; 2) automatically estimates their relative positions and orientations; and 3) shows robust *real-time* performance, thanks to its low complexity and distributed computation load.

C. Radar Networks Self-Calibration

Few works have tackled the problem of self-calibration in mmWave radar networks, that is, [10], [11], and [26]. These studies share significant practical limitations. Copa et al. [26] proposes a semiautomatic calibration in which a set of reflectors (e.g., oscillating pendula) has to be deployed in the environment and then removed once calibration is complete. Moreover, only the relative rotation between radars is estimated, while the translations are manually measured. Li et al. [10], instead, performed calibration using a target of opportunity, but it requires the presence of only one target following a strictly *linear* trajectory in the radars field of view (FoV). Finally, Iwata et al. [11] can handle multiple subjects, but all of them need to be static (e.g., sitting). Such assumptions considerably limit the application scope of these systems. Conversely, our method completely automates the calibration process, working with movement trajectories of arbitrary shapes and with multiple concurrently moving targets. Actually, ORACLE *benefits* from having multiple trajectories of complex and irregular shapes that span a large portion of the FoVs of the radars, as they lead to a more accurate calibration.

III. PROBLEM OUTLINE

In this section, we first present an overview of mmWave multiple-input multiple-output (MIMO) radars. Then, we formalize the problems of combining the information obtained by the different radars in the network at a central fusion entity and estimating their relative positions and orientations.

A. mmWave MIMO Radars

An MIMO frequency-modulated continuous-wave (FMCW) radar jointly estimates the distance, the radial velocity, and the angular position of the targets with respect to the radar itself [27]. During the sensing process, the radar transmits sequences of linear chirp signals with bandwidth $\mathcal{B}$. A full sequence, or "radar frame," is repeated with a period of T_s seconds. The distance, r, and velocity, v, of the targets are computed from the frequency shift induced by the delay of each reflection, usually by applying discrete Fourier transform (DFT) processing. The FMCW radar distance resolution is related to the bandwidth $\mathcal{B}$ by $\Delta r = c/(2\mathcal{B})$, where c is the speed of light. This makes mmWave devices accurate to the level of a few centimeters using a bandwidth of

2–4 GHz [4]. Furthermore, using a 2-D array of multiple receiving antennas makes it possible to obtain the angle of arrival (AoA) of the reflections along the azimuth (θ) and the elevation (ϕ) domains, by leveraging phase shifts across different antenna elements. The azimuthal AoA resolution depends on the number of antennas N in the array and is given by $\Delta\theta = \lambda/(Nd\cos\theta)$, where d is the spacing between the antennas. Due to the high-ranging resolution, a human presence in the environment generates a large number of reflecting points, which are detected by the radar. This set of points, usually termed *radar point cloud*, can be transformed into the 3-D Cartesian space using the distance, azimuth, and elevation angles information of the multiple body parts. Each point is described by a vector $[x, y, z]^T$ including the point's spatial coordinates x, y, z obtained by transforming r, θ, and ϕ. Movement trajectories can be tracked across time from the point clouds.

B. Sensor Fusion in mmWave Radar Networks

Consider an mmWave radar network consisting of S monostatic radar sensors. Each radar has local computational capabilities and a communication interface that enables them to transmit information to an FC. The sensors are identified by indices $s = 1, \ldots, S$, while quantities related to the FC are denoted by superscript ^{c}. All radar sensors operate at discrete time steps of duration T_s, indexed by variable k. The FC also operates at discrete time steps that, in general, may have a different duration T_c and are indexed by variable m.

The people-tracking problem relates to estimating the subjects' movement trajectories in the (x, y) horizontal plane across time, exploiting the measurements of the multiple radar sensors. For this, we define the *state* of subject u, seen by the FC at time m, as $\mathbf{x}_m(u) = [x_m(u), y_m(u), \dot{x}_m(u), \dot{y}_m(u)]^T$, containing u's coordinates and the corresponding velocity components $\dot{x}_m(u)$ and $\dot{y}_m(u)$. We assume that the state's evolution obeys a constant-velocity (CV) model [28]. At the FC, the state model for target u is

$$\mathbf{x}_m(u) = \mathbf{F}_{T_c}\mathbf{x}_{m-1}(u) + \mathbf{w}_m(u) \tag{1}$$

where $\mathbf{F}_{T_c}$ is the state transition matrix that projects the state forward by a time duration T_c, according to the CV model, while $\mathbf{w}_m(u)$ is the (global) Gaussian process noise, having zero-mean and covariance matrix $\mathbf{W}$ [4], [29]. The process noise is here considered to be generated by a random acceleration that is not explicitly accounted for by the CV model [4]. Sensor measurements of the state of target n, at time k, are obtained according to

$$\mathbf{z}_k^s(n) = \mathbf{H}\mathbf{x}_k(n) + \mathbf{v}_k^s(n), \quad s = 1, \ldots, S \tag{2}$$

where $\mathbf{z}_k^s(n)$ is the observation obtained from sensor s, $\mathbf{H}$ is the observation matrix relating the observation to the state, and $\mathbf{v}_k^s(n)$ is the sensor-specific measurement noise having covariance matrix $\mathbf{V}_k$ [4]. In our system, all sensors are of the same type and have the same specifications. Therefore, we can safely assume that the measurement error processes have the same zero-mean, Gaussian distribution, whose covariance is time-varying due to the dependence of the radars' resolution on the position of the targets in the FoV [4].

Our system aims to estimate $\mathbf{x}_m$ over time, exploiting the measurements collected by the sensors. Note that: 1) the correspondence between the targets tracked by each sensor is *unknown*, and finding a suitable association between them is part of the problem we tackle; 2) the algorithm can handle the fact that the same target may be tracked simultaneously by one or more sensors; and 3) the sensors may collect the measurements and obtain state estimates at a different time granularity than that of the FC.

Each sensor locally tracks the targets in the environment. The common approach to people tracking from mmWave radar point clouds [4], [5], [25] includes: 1) a detection phase via density-based clustering algorithms (e.g., density-based spatial clustering of applications with noise (DBSCAN) [30]) to separate the reflections from multiple subjects and 2) applying Kalman filtering techniques [31] on each cluster centroid to track the movement trajectory of each subject in space. The association between new observations and Kalman filter (KF) states exploits the nearest-neighbors joint probabilistic data filter (NN-JPDAF) algorithm [32]. The KF used at each sensor provides, at each time step, an estimate of the state of the targets in its FoV and the corresponding error covariance. We call N_k^s the number of such targets at time k, $\hat{\mathbf{x}}_{k|k}^s(n)$ the estimated state of the target n after the KF update step, and $\mathbf{C}_{k|k}^s(n)$ the associated error covariance. Note that the above quantities are sensor-dependent, as different sensors provide their own estimates of the state of the same target. We denote by $\mathcal{T}_k^s(n) = \{\hat{\mathbf{x}}_{k|k}^s(n), \mathbf{C}_{k|k}^s(n)\}$ the track corresponding to target n as estimated by sensor s, expressed with respect to *its own* reference frame. In addition, we assume that each sensor can provide a timestamp, τ_k^s, corresponding to the current time step, according to its local time reference (e.g., internal clock or network time). For the timestamps to match between different sensors, some level of synchronization is needed within the radar network (e.g., using NTP). At the end of each time step, sensor s transmits the set of its tracks, denoted by $\mathcal{T}_k^s = \{\mathcal{T}_k^s(1), \ldots, \mathcal{T}_k^s(N_k^s), \tau_k^s\}$ to the FC, together with the corresponding timestamp.

If the same target is tracked by more than one sensor, the FC should maintain a single track for it, which is updated and improved by fusing the information coming from the sensors. We aim to develop an algorithm to estimate the position of the targets across time at the FC, in the form of *central tracks* $\mathcal{T}_m^c(u) = \{\hat{\mathbf{x}}_{m|m}^c(u), \mathbf{C}_{m|m}^c(u)\}$, obtained by combining the sensor information $\mathcal{T}_k^s$, $s = 1, \ldots, S$. The above problem is complicated by *correlation* between the estimation errors of the tracks obtained at the sensors and the FC. From (1), one can see that some correlation exists between all the tracks that refer to the same target (also in different sensors), as the process noise is the same, but this can be typically neglected if the process noise has low intensity or if the radar measurement rate is high with respect to the subject's motion [29]. Conversely, the error correlation between a central track and a sensor track of the same target cannot be ignored, as the FC obtains its own tracks as a function of the sensor tracks. This is especially true for our real-time application, where the fusion occurs frequently, for example, from 10 to 15 times per second.

C. Self-Calibration of mmWave Radar Networks

The track sets that the radars transmit to the FC are expressed in the local reference frames of the sensors. Any algorithm that fuses them to improve the tracking accuracy requires to know the sensors' relative position and orientation. However, manually measuring them is impractical and prone to errors, and therefore, an automatic self-calibration procedure is highly appealing. Here, we propose to exploit the trajectories of *targets of opportunity* that move within the radars' FoVs, independently tracked by each sensor. Tracks from different radars that correspond to the same target have almost the same *shape*, up to a rigid transformation and some noise. Estimating such rigid transformation parameters corresponds to estimating the sensors' relative position and orientation.

Considering the system of S radars, deployed in the same area, call $\mathcal{F}_s$, $s = 1, \ldots, S$, their reference systems (RSs). Each RS consists of a pair $\mathcal{F}_s = \{\mathbf{t}_s, \mathbf{R}_s\}$, where $\mathbf{t}_s$ is the 2×1 vector with the coordinates of the sth RS's origin and $\mathbf{R}_s$ is the 2×2 rotation matrix specifying its orientation. Without loss of generality, in this article, we consider a *global* RS (of the FC) which coincides with that of radar 1 and for which it holds that $\mathbf{t}_1 = \mathbf{0}_{2\times1}$ and $\mathbf{R}_1 = \mathbf{I}_2$, respectively, the 2×1 zero vector and the rank 2 identity matrix. Self-calibrating the system consists in estimating $\mathcal{F}_s$, $s = 2, \ldots, S$. We define the *movement trajectory* of target n, as seen by sensor s, as the sequence of position estimates of the target, $\hat{\mathbf{p}}_k^s(n) = [\hat{x}_k^s(n), \hat{y}_k^s(n)]^T$, for $k = 1, \ldots, K$, where k is the discrete time index. Note that $\hat{\mathbf{p}}_k^s(n)$ contains the first two components of the KF state estimate $\hat{\mathbf{x}}_{k|k}^s(n)$. An estimate of the rotation matrix and the translation vector between radar s and radar 1 (our reference) can be obtained by solving the following least-squares (LS) problem:

$$\underset{\substack{\mathbf{R}_s \in \mathrm{SO}(2) \\ \mathbf{t}_s \in \mathbb{R}^2}}{\arg\min} \sum_{k=1}^{K} \left\| (\mathbf{R}_s \hat{\mathbf{p}}_k^s(n) + \mathbf{t}_s) - \hat{\mathbf{p}}_k^1(n) \right\|_2 \qquad (3)$$

where SO(2) denotes the special orthogonal group in dimension 2 (i.e., the set of all possible rotations around a point in a 2-D space) and $\|\cdot\|_2$ is the Euclidean norm. While the translation vector of sensor s with respect to the global RS is directly obtained by solving (3), the orientation angle, denoted by θ_s, is given by $\theta_s = \cos^{-1}(\mathrm{trace}(\mathbf{R}_s)/2)$.

In Section IV-C, we present the proposed approach to solve the self-calibration problem in the more complex and realistic scenario where: 1) multiple sensors concurrently track multiple targets; 2) the track–target correspondence among different sensors is *unknown*, so, an association strategy has to be developed; and 3) the trajectories should be aligned in time before using (3).

IV. PROPOSED APPROACH

In this section, we first present a high-level overview of the processing blocks of ORACLE and then provide a detailed description of each of them. Fig. 1 presents the workflow of ORACLE.
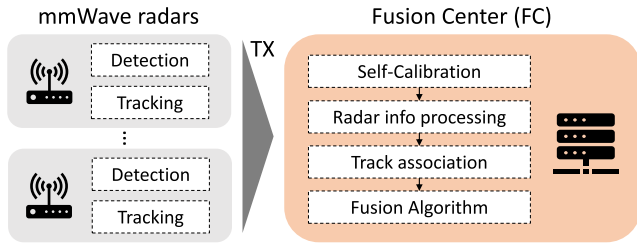
Fig. 1. Proposed workflow.

A. Self-Calibration

In this phase, the relative positions and orientations of the radars are obtained from the trajectories of targets of opportunity (see Section IV-C). The steps are as follows.

1) *Time Alignment:* A time alignment between the trajectories from radar 1 and the trajectories from radar s is sought, by minimizing the distance between the trajectories' timestamps (see Section IV-C1).

2) *Track Association:* Using the time alignment from point 1), we solve the problem in (3) for all the trajectory pairs and compute a corresponding association cost matrix. Using the cost matrix, the best associations between track pairs are computed (see Section IV-C2).

3) *Masking:* When estimating the roto-translation parameters at point 4), multiple track pairs will be used all together. To avoid possibly wrong associations spoiling the final results, all possible subsets of the best associations from point 2) are considered through a *masking* operation, and a new association cost is computed using all the trajectories in each subset. In the end, the subset giving the lowest cost will be selected for the final parameters estimation in point 4) (see Section IV-C3).

4) *Radar Calibration:* All the track pairs from point 3) are stacked together and used to set up a rigid transformation problem as in (3) that provides the final position and orientation estimates for the radar (see Section IV-C4).

B. Multiradar Fusion

Here, the tracks from the radars are fused at the FC to build a set of *central* tracks associated with the subjects in the environment (see Section IV-D). This includes the following.

1) *Slotted Sensor Information Processing:* The track information from the sensors is sent to the FC and processed using a slotted protocol (see Section IV-D1).

2) *Track Association:* A method to associate (frame-by-frame) those tracks that correspond to the same target, according to their statistical similarity, is used to select pairs of tracks to be fused (see Section IV-D2).

3) *Radar Track Fusion Algorithm:* The fusion algorithm combines sensor tracks with the central tracks using different rules depending on the type of fusion event (see Section IV-D3).

C. Self-Calibration

1) Time Alignment: For simplicity of notation, call $\mathbf{n}_1$ a trajectory from sensor 1 and $\mathbf{n}_s$ a trajectory from sensor s.

Each of them contains the sequence of the position estimates of some target that may, or may not, coincide. Sensors communicate position estimates to the FC along with their timestamps. Note that the trajectories may have a different length. The time alignment is then performed so that each position estimate of trajectory $\mathbf{n}_1$ is associated with the position estimate of trajectory $\mathbf{n}_s$ that minimizes the time difference between the two acquisition instants. Elements of trajectory $\mathbf{n}_1$ that do not have a corresponding element of trajectory $\mathbf{n}_s$ within T_c seconds are discarded and vice versa (recall that T_c is the duration of an FC time step). This operation reduces the trajectories to a common length of K time-aligned positions. Call $\mathbf{k}_1$ and $\mathbf{k}_s$ the vectors containing the indices that provide the time-aligned sequences from radars 1 and s, respectively. Note that $\mathbf{k}_1$ and $\mathbf{k}_s$ have the same length. With the time alignment operation, we retain only the portions of the trajectories that are sufficiently well synchronized, to avoid performing the rigid transformation on wrongly associated points. Once the trajectory association has been established, we define the *mean time shift* of the pair $\{\mathbf{n}_1, \mathbf{n}_s\}$ as $\bar{\tau}(\mathbf{n}_1, \mathbf{n}_s) = (1/K) \sum_{k=1}^{K} |\tau_{\mathbf{k}_{1,k}}^1 - \tau_{\mathbf{k}_{s,k}}^s|$, where $\mathbf{k}_{j,k}$ denotes the kth element of vector $\mathbf{k}_j$. The value of $\bar{\tau}(\mathbf{n}_1, \mathbf{n}_s)$, expressed in seconds, is related to the alignment quality of the two trajectories and will be used in the association step (see Section IV-C2).

2) Track Association: Our data association strategy consists of computing a *cost* for each pair $\{\mathbf{n}_1, \mathbf{n}_s\}$ and solving the resulting combinatorial cost minimization problem to obtain the best associations. We assume to have N_1 and N_s trajectories available at radars 1 and s, respectively. Our cost function incorporates different aspects: 1) the length of the trajectories, as longer trajectories are assumed to provide a better calibration; 2) the time alignment of the trajectories, as we should compare position estimates acquired almost simultaneously by the different radars; and 3) the quality of the rigid transformation, in terms of residual error in superimposing trajectories from the different radars. We define the association cost, A, for the pair $\{\mathbf{n}_1, \mathbf{n}_s\}$, as

$$A(\mathbf{n}_1, \mathbf{n}_s) = -\rho(K, \bar{\tau}) \left[1 + \xi(\mathbf{n}_1, \mathbf{n}_s)\right]^{-1} \quad (4)$$

where $\xi(\mathbf{n}_1, \mathbf{n}_s)$ is the sum of the LS residuals, after applying the time alignment and the rigid transformation, while $\rho(K, \bar{\tau})$ is a factor that favors trajectory pairs with a long overlap and a low mean time shift. The rigid transformation parameters $\mathbf{R}_s^{(\mathbf{n}_1, \mathbf{n}_s)}, \mathbf{t}_s^{(\mathbf{n}_1, \mathbf{n}_s)}$ are computed, using the time-aligned track pairs from point 1), solving the LS problem in (3) in closed-form through a singular-value decomposition (SVD) method [33]. Then, the LS residuals sum is computed as $\xi(\mathbf{n}_1, \mathbf{n}_s) = \sum_{k=1}^{K} ||\mathbf{n}_{1,k} - \mathbf{R}_s^{(\mathbf{n}_1, \mathbf{n}_s)} \mathbf{n}_{s,k} - \mathbf{t}_s^{(\mathbf{n}_1, \mathbf{n}_s)}||_2$, where k indexes trajectory positions. Recalling that T_c is the sampling interval of the FC, the corrective term is formalized as $\rho(K, \bar{\tau}) = \ln(KT_c)[1 + \bar{\tau}(\mathbf{n}_1, \mathbf{n}_s)]^{-1}$. Costs $A(\mathbf{n}_1, \mathbf{n}_s)$ are arranged into an $N_1 \times N_s$ cost matrix and the optimal association of trajectories is obtained by minimizing the overall cost, computed through the Hungarian algorithm [34]. This yields $N_t = \min(N_1, N_s)$ pairs of associated trajectories, which are possibly the same targets seen by the two radars. Due to the presence of spurious trajectories, ghost targets, and clutter, we select a subset of the associated trajectory pairs

that have a cost below a threshold A_{self}, which represents a confidence value under which the pair is truly a trajectory pair generated by a human. Denote the set of selected track pairs by $Q^{1s} = \{\{\mathbf{s}_1^1, \mathbf{s}_1^s\}, \ldots, \{\mathbf{s}_{N_t}^1, \mathbf{s}_{N_t}^s\}\}$, where $\mathbf{s}_p^q$ indicates the pth selected track from radar q. In our experiments, we empirically adopted $A_{\text{self}} = 18$.

3) Masking: In phase 4) (see Section IV-C4), one or more of the N_t track pairs selected during the previous phase are used combinedly to compute the final self-calibration parameters. Ideally, each of the selected track pairs should provide two trajectories, one from each radar, corresponding to the same target. However, in practice, there might be wrong associations. To mitigate this shortcoming, in phase 3), all possible subsets of the selected N_t tracks are considered, through a *masking* operation on the one-to-one track associations. We call it a masking operation as it corresponds to purposely ignoring (*masking*) some of the track associations in the computation of the self-calibration parameters. The same association cost of (4) is computed by stacking together all the trajectories in each subset. Then, the subset providing the lowest cost is used for the final calibration. Formally, let $P(Q^{1s})$ be the set of all possible subsets of Q^{1s} excluding the empty set. Recall that Q^{1s} is the set of all selected track pairs after the track association phase. Each element of $P(Q^{1s})$ is a set of track pairs from radar 1 and radar s. For each element of $P(Q^{1s})$, all the trajectories from sensor 1 are stacked in vector $\mathbf{q}_1$ and all the trajectories from sensor s are stacked in vector $\mathbf{q}_s$ and the same operation is performed with the corresponding timestamp sequences. Then, cost $A(\mathbf{q}_1, \mathbf{q}_s)$ is computed as in (4) and all costs are stored in a matrix of dimension $(2^{N_t} - 1) \times 1$. The element of $P(Q^{1s})$ providing the lowest cost is selected. The $N_t^* \leq N_t$ trajectory pairs contained in such minimum-cost elements are used in phase 4) to compute the self-calibration parameters. Since the masking phase cost is exponential in the number of track-to-track associations, it is possible to limit the maximum number of track pairs to be retained from Q^{1s}. In this case, the track pairs with the highest cost are to be excluded. In our experiments, we used a maximum of five track pairs.

4) Radar Calibration: The N_t^* trajectory pairs selected during the masking phase are then stacked together and used to set up a rigid transformation problem as in (3). The problem is solved with the same procedure described in Section IV-C2 [33], obtaining the final rotation matrix and translation vector to calibrate radar s, namely $\{\mathbf{R}_s^*, \mathbf{t}_s^*\}$. This step exploits all the available information from multiple subjects, improving the calibration accuracy by increasing the number of useful measurements per time frame. Note that, even though target occlusion events may split a trajectory into multiple components, our algorithm still works by exploiting each resulting subtrajectory.

D. Multiradar Fusion

Once the radar network is calibrated, that is, we have an estimate $\{\mathbf{R}_s^*, \mathbf{t}_s^*\}, \forall s$, we can fuse the information coming from the S radars at the FC. In the following, we consider $S = 2$ for better clarity in the algorithm description, but the method works for an arbitrary S. We denote the *precision matrix*,
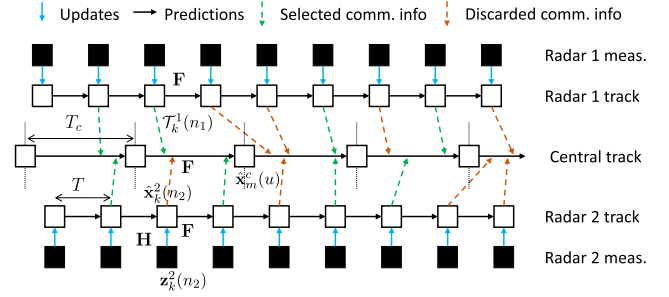

Fig. 2. Scheme of the proposed fusion algorithm with two radars.

which is defined as the inverse of a covariance matrix, by $\mathbf{P} = \mathbf{C}^{-1}$. The fusion algorithm, represented in Fig. 2, is described as follows.

1) Slotted Sensor Information Processing: The FC maintains a central time variable, denoted by $\tau_m^c = \tau_0 + mT_c$, which is incremented at the end of each central time step and where τ_0 is the time when the FC starts operating. To cope with the random variations in the sensor acquisition, processing, and communication times, the FC operates on time slots of duration T_c. Specifically, several track sets from different sensors can be received during time step m due to differences between the FC and the sensors' time steps and the variable communication time. Using the timestamp information contained in the received tracks, at time m, the FC filters out all the track sets that are not received within the interval $(\tau_{m-1}^c, \tau_m^c]$ and retains only the most recent track set from each sensor. Formally, for each s, we select the track set whose timestamp is the solution to $\arg\min_{\tau_k^s}(|\tau_m^c - \tau_k^s|)$. In the following, to highlight that a track set has been selected from sensor s to be processed in time step m, we denote it as $\mathcal{T}_m^s$, using the time index of the FC rather than that of the sensor, and we do the same for all the tracks it contains.

The slotted processing procedure: 1) reduces the number of fusion steps the FC carries out, using only the most recent information available from each sensor and 2) avoids erroneously fusing outdated tracks.

After selecting the sensor tracks, the FC transforms them to match its own RS, using the information about the location and orientation of the radar sensors. Then, according to the CV model, the tracks are propagated to the FC current time. We denote by $\bar{\mathbf{t}}_s^* = [\mathbf{t}_s^*, \mathbf{0}_{1\times 2}]^T$ the augmented translation vector and by $\bar{\mathbf{R}}_s^* = \text{blkdiag}(\mathbf{R}_s^*, \mathbf{R}_s^*)$ the 4×4 augmented rotation matrix of sensor s, where $\text{blkdiag}(\cdot)$ returns a block diagonal matrix of its inputs. The transformation and propagation are performed, together, as

$$\hat{\mathbf{x}}_m^s(n) = \mathbf{F}_{\tau_m^c - \tau_k^s}\left[\bar{\mathbf{R}}_s^* \hat{\mathbf{x}}_m^{s'}(n) + \bar{\mathbf{t}}_s^*\right] \tag{5}$$

$$\mathbf{C}_m^s(n) = \mathbf{F}_{\tau_m^c - \tau_k^s} \bar{\mathbf{R}}_s^* \mathbf{C}_m^{s'}(n) \left(\bar{\mathbf{R}}_s^*\right)^T \mathbf{F}_{\tau_m^c - \tau_k^s}^T \tag{6}$$

with $\hat{\mathbf{x}}_m^{s'}(n)$ and $\mathbf{C}_m^{s'}(n)$ being the state and covariance communicated by sensor s that have been selected in the current central slot, expressed in the reference frame of sensor s, while $\hat{\mathbf{x}}_m^s(n)$ and $\mathbf{C}_m^s(n)$ are expressed in the reference frame of the FC. In (5) and (6), the state evolution matrix $\mathbf{F}_{\tau_m^c - \tau_k^s}$

projects the sensor state/covariance estimates forward by $\tau_m^c - \tau_k^s$, so that they are up to date with the current FC time. Similarly, the FC also performs a prediction step, for a time duration T_c, on all its maintained tracks, by leveraging their motion model. For this, the standard KF prediction equations are used

$$\mathbf{C}_{m|m-1}^c(u) = \mathbf{F}_{T_c} \mathbf{C}_{m-1|m-1}^c(u) \mathbf{F}_{T_c}^T + \mathbf{W} \tag{7}$$

$$\hat{\mathbf{x}}_{m|m-1}^c(u) = \mathbf{F}_{T_c} \hat{\mathbf{x}}_{m-1|m-1}^c(u). \tag{8}$$

2) Track Association: The FC has to compute track-to-track associations before being able to fuse the information from the sensors with its own tracks, as it needs to identify which tracks correspond to the same target. There can be: 1) *sensor-to-center* associations (SC), to verify whether the sensor tracks correspond to any of the maintained central tracks and 2) *sensor-to-sensor* associations (SS), only for sensor tracks which did not find an SC association, to establish which tracks correspond to the same targets and consequently initialize the correct number of central tracks. The aim is to find a one-to-one association between two sets of tracks, respectively, indicated by variables i and j. Note that i and j may refer to two sensor tracks, in the case of an SS association, or to a central track and a sensor track, in the case of an SC association.

Initially, SC associations are considered. As the first step, it is verified whether associations from previous time steps are still valid. To this purpose, unique identifiers associated with each sensor, central track, and sensor track are exploited. Every SC track pair that has correspondence in previous SC associations is examined to verify whether the association still holds. This operation consists of computing the Mahalanobis distance [32] between the two tracks and confirming the association only if the distance is lower than, or equal to, a threshold A_{th}. Formally, it is computed as

$$d_M(i, j) = \left(\hat{\mathbf{x}}(i) - \hat{\mathbf{x}}(j)\right)^T \mathbf{P}(i, j) \left(\hat{\mathbf{x}}(i) - \hat{\mathbf{x}}(j)\right) \tag{9}$$

$$\mathbf{P}(i, j) = [\mathbf{C}(i) + \mathbf{C}(j)]^{-1} \tag{10}$$

where $\mathbf{P}(i, j)$ is the precision matrix inducing the distance. All track pairs for which $d_M(i, j) \leq A_{\text{th}}$ are retained as valid SC associations, while the remaining ones undergo the following further association stages. Assume M and N central and sensor tracks are left to be associated, respectively. An $M \times N$ cost matrix, Λ, is obtained, where the value of entry Λ_{ij} is computed differently depending on the relationship between central track i and sensor track j.

If i and j were previously fused together within a time interval of T_{th} seconds, then, Λ_{ij} is computed as in (9) with the difference that each track state estimate $\hat{\mathbf{x}}(q)$ and each covariance matrix $\mathbf{C}(q)$, $q = i, j$, is replaced by $\hat{\mathbf{x}}_{\text{dec}}(q) = \hat{\mathbf{x}}(q) - \bar{\mathbf{x}}(j)$ and $\mathbf{C}_{\text{dec}}(q) = (\mathbf{C}(q)^{-1} - \bar{\mathbf{C}}(j)^{-1})^{-1}$, respectively. $\bar{\mathbf{x}}(j)$ and $\bar{\mathbf{C}}(j)$ are the last communicated state and covariance matrix from track j, respectively. $\hat{\mathbf{x}}_{\text{dec}}(q)$ and $\mathbf{C}_{\text{dec}}(q)$ represent the decorrelated versions of the corresponding quantities, according to the *decorrelation principle* [29], [35]. The decorrelation operation removes the effect of previous fusion events that, otherwise, would affect the computation of the association cost [29].

If i and j were never fused together, or the fusion event happened more than T_{th} seconds before, then, $\Lambda_{ij} = d_M(i, j)$, as in (9), without modifications. Once matrix Λ is available, the minimum total cost association is obtained by using, for example, the Hungarian algorithm [34], and all associations whose cost does not exceed threshold A_{th} are considered as valid SC associations.

After these operations, all acceptable SC associations have been established and only SS associations are left to be computed. Let i and j be two tracks from different sensors. Then, a similar cost matrix Λ is built, where $\Lambda_{ij} = d_M(i, j)$, as in (9). Since sensor tracks originated from different sensors, they are negligibly correlated and there is no need to apply any decorrelation operation on them. Then, the minimum total cost association is obtained, as before, using, for example, the Hungarian algorithm [34]. In our experiments, we adopted $A_{\text{th}} = 18$ and $T_{\text{th}} = 1.3 \times T_c$.

3) Radar Track Fusion Algorithm: The track fusion algorithm behaves differently in case it has to combine two sensor tracks (SS fusion) or one sensor track with a central track (SC fusion). If the FC is currently not maintaining any track for a certain subject, but one or more radars are, a new central track needs to be initialized based on the received information from the local sensors. Specifically, two cases may happen: 1) if a target n_1 is currently tracked by one sensor only, the corresponding central track is initialized using the state and covariance of the sensor track and 2) if the FC receives two tracks that can be associated, say $\mathcal{T}_m^1(n_1)$ and $\mathcal{T}_m^2(n_2)$, these are fused into a single, new track associated with target with central index u (SS fusion). The local tracks from sensors 1 and 2 have uncorrelated (or negligibly correlated) errors as they are two sensor tracks, so they can be fused with a weighted combination of their states [29]. The states are weighted by the precision matrices associated with the estimation errors at each sensor. The fusion equations used for the initialization of a new central track, at time m, in case 2), are

$$\mathbf{C}_{m|m}^c(u) = \left[\mathbf{P}_m^1(n_1) + \mathbf{P}_m^2(n_2)\right]^{-1} \tag{11}$$

$$\hat{\mathbf{x}}_{m|m}^c(u) = \mathbf{C}_{m|m}^c(u) \left[\mathbf{P}_m^1(n_1)\hat{\mathbf{x}}_m^1(n_1) + \mathbf{P}_m^2(n_2)\hat{\mathbf{x}}_m^2(n_2)\right] \tag{12}$$

for the couple of associated tracks $\mathcal{T}_m^1(n_1)$ and $\mathcal{T}_m^2(n_2)$. Note that, to detect when an SS fusion has to be performed, our algorithm applies the SS association procedure to all the sensor tracks that have *not* been associated with any central track in the current slot. In case more than two sensors are available, the above process is repeated sequentially using sensors 1 and 2 first, then fusing the resulting track with the information from sensor 3 and so on until the track sets of all S sensors are used.

On the other hand, if the FC has already initialized the track for a subject, the fusion has to be performed between the central track and a sensor track corresponding to the same subject. Denote by $\mathcal{T}_m^c$ the set of tracks maintained by the FC at the central time step m. Upon receiving the local information $\mathcal{T}_m^s$, the FC runs the track-to-track association algorithm

to find pairs of corresponding tracks $\{\mathcal{T}_m^c(u), \mathcal{T}_m^s(n)\}$. Once such pairs are available, if the timestamp associated with $\mathcal{T}_m^s$ is older than T_{th}, then, the same fusion rule of (11) and (12) is used, as the two tracks can be considered sufficiently decorrelated, otherwise, each central track is updated with its corresponding sensor track using the *information decorrelation method* [29], [35] as follows:

$$\mathbf{C}_{m|m}^c(u) = \left[\mathbf{P}_{m|m-1}^c(u) + \mathbf{P}_m^s(n) - \bar{\mathbf{P}}^s(n)\right]^{-1} \quad (13)$$

$$\hat{\mathbf{x}}_{m|m}^c(u) = \mathbf{C}_{m|m}^c(u)\left[\mathbf{P}_{m|m-1}^c(u)\hat{\mathbf{x}}_{m|m-1}^c(u)\right.$$
$$\left. + \mathbf{P}_m^s(n)\hat{\mathbf{x}}_m^s(n) - \bar{\mathbf{P}}^s(n)\bar{\mathbf{x}}^s(n)\right] \quad (14)$$

where $\bar{\mathbf{P}}^s(n)$ and $\bar{\mathbf{x}}^s(n)$ are the last communicated precision matrix and state estimate of track $\mathcal{T}_m^s(n)$ from sensor s to the FC, respectively. Information decorrelation copes with the problem of time-correlated tracks between the FC and the radar sensors, by removing the most recently received information about target n (or u from the FC perspective), as, otherwise, this would be accounted for twice.

4) Track Initialization and Termination: To deal with the initialization and termination of central tracks, while keeping the complexity of the system as low as possible, we follow a so-called m/n logic, similar to what is done for the local tracking process of each radar sensor [4]. Specifically, a track is maintained if it is associated with any of the received sensor tracks for at least m out of the last n frames. Similarly, received sensor tracks that are not associated with any existing central track are initialized as new tracks if they are detected for at least m out of the last n frames. As detailed in Section IV-D3, before initializing a new central track, the received selected tracks form the radars are associated and fused with an SS fusion step, whenever possible. In this way, we avoid multiple initializations of the tracks corresponding to the same targets.

We remark that the proposed track initialization and termination strategy allows ORACLE to robustly handle people entering and leaving the FoV of the radar network. Whenever a person walks inside the FoV of one radar, the latter starts tracking the person and sending the tracks to the FC. The same applies to the other radars in the network and all the received tracks pertaining to the same subject are fused together. Therefore, as long as there is at least one radar tracking a subject, their track is also maintained at the FC. This allows nimbly handling cases when a person moves across different radar FoVs, concurrently leaving and entering the FoVs of different radar devices, by always maintaining a valid track for the user at the fusion point.

V. EXPERIMENTAL RESULTS

We evaluated ORACLE through implementation of the RadNet platform [14] featuring four Texas Instruments IWR1843BOOST mmWave radars[1] connected to two Jetson Xavier NX DevKit,[2] one Jetson TX2 DevKit[3] and one Jetson Nano DevKit,[4] all communicating via Ethernet. The radars

operate in the 77–81 GHz band in real-time at a frame rate of $1/T_s = 15$ Hz with an FoV of $\pm 60°$ and $\pm 15°$ over the azimuth and elevation planes, respectively.

A. Implementation Notes on Numerical Stability

In this section, we provide insights into the numerical stability of ORACLE, which is key to implementing the system in practice. Specifically, during our experiments, we observed that ORACLE's operations on covariance matrices (e.g., the information decorrelation or roto-translations) can easily make them: 1) nonpositive definite and 2) ill-conditioned (with very large condition numbers), causing wrong track associations and fusion results because of the spoiled inverse matrices. The solution to point 1) is to enforce positive definiteness by adding a properly designed diagonal matrix. Formally, let $\mathbf{P}$ be a symmetric square matrix obtained by ORACLE as part of the information fusion process, and $\lambda_{min} \leq 0$ be its minimum, nonpositive eigenvalue. We obtain the corrected, positive definite matrix $\mathbf{P}_{pos}$, as

$$\mathbf{P}_{pos} = \mathbf{P} + (-\lambda_{min} + \varepsilon)\mathbf{I}, \quad \varepsilon > 0. \quad (15)$$

This approach relies on the fact that a symmetric matrix is positive definite if and only if all its eigenvalues are positive. As a solution for 2), we use ridge regularization [36] to limit the condition number of the matrix. Let $\lambda'_{min} > 0$ and $\lambda'_{max} > 0$ be, respectively, the minimum and maximum eigenvalues of the positive definite matrix $\mathbf{P}_{pos}$. Denote by $\text{cond}(\mathbf{P}_{pos}) = \lambda'_{max}/\lambda'_{min}$ the condition number of matrix $\mathbf{P}_{pos}$, and by δ the regularization parameter. The regularized covariance matrix, $\mathbf{P}_{reg}$, is obtained as

$$\mathbf{P}_{reg} = \frac{1}{1+\delta}(\mathbf{P}_{pos} + \delta\mathbf{I}). \quad (16)$$

As a result of (16), the minimum and maximum eigenvalues of $\mathbf{P}_{reg}$ are $\lambda'_{max} + \delta$ and $\lambda'_{min} + \delta$, respectively. To limit the condition number of $\mathbf{P}_{reg}$, we specify an upper bound for its value, denoted by c^*. Then, such bound is enforced by computing δ from $\text{cond}(\mathbf{P}_{reg}) = (\lambda'_{max} + \delta)/(\lambda'_{min} + \delta) \leq c^*$, which is solved with equality by

$$\delta = \max\left(0, \frac{\lambda'_{max} - c\lambda'_{min}}{c^* - 1}\right). \quad (17)$$

In our experiments, we empirically decided to adopt $c^* = 50$. ORACLE applies the correction in (15) whenever a covariance (or precision) matrix is nonpositive definite. The regularization in (16), instead, is used if the condition number of a covariance (or precision) matrix is above c^*.

B. Measurements Setup and Dataset

To assess the performance of the proposed method, we conducted tests in a $7 \times 4\,\text{m}^2$ research laboratory [see Fig. 3(a)] equipped with a motion tracking system featuring ten cameras. This provides the ground-truth (GT) 3-D localization of a set of markers placed on the radars and the moving subjects with millimeter-level accuracy. We considered two different scenarios with four radars and 1, 2, and 3 moving targets. Fig. 3 shows the locations and orientations of the radars in the

---

[1]https://www.ti.com/tool/IWR1843BOOST

[2]https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit

[3]https://developer.nvidia.com/embedded/buy/jetson-tx2-developer-kit

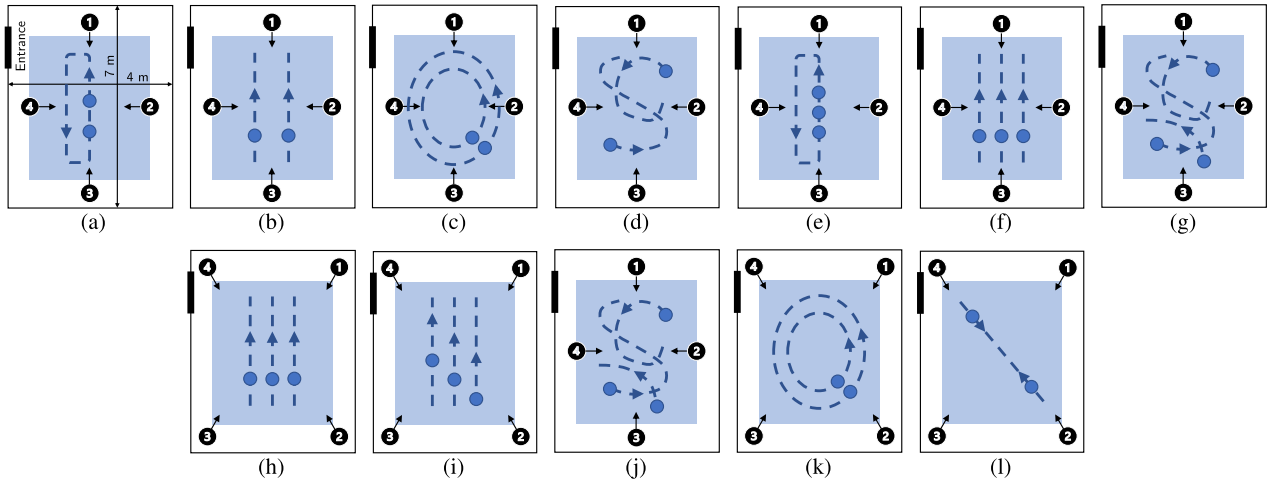[4]https://developer.nvidia.com/embedded/buy/jetson-nano-devkit

Fig. 3. Setup schemes. The black numbered dots represent the radar devices and the arrows identify their pointing direction. The blue dots represent the moving people, while the dashed lines show the traveled trajectories in the direction given by the blue arrows. The first row shows setup-1 deployments, whereas the second row shows setup-2 deployments. (a) In-line, (b) parallel, (c) circular, (d) free, (e) in-line, (f) parallel, (g) free, (h) parallel, (i) paral-diag, (j) free, (k) circular, and (l) vs-in-line.
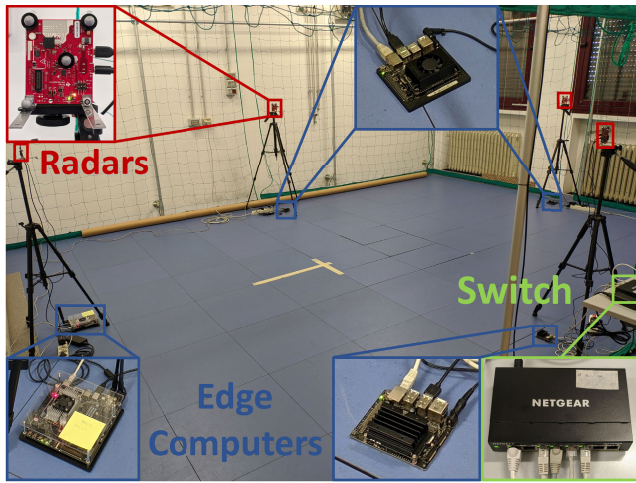

Fig. 4. Experimental setup. The FC, not shown in the picture, is connected to the edge computers through a switch.

different setups, where the black numbered dots represent the radar devices and the arrows identify their pointing direction. The blue dots represent the moving people, while the dashed lines show the traveled trajectories in the direction given by the blue arrows. The first row shows setup-1 deployments whereas the second row shows setup-2 deployments. We also asked the subjects to move according to six possible different trajectories: 1) *in-line*, identifying a movement along a straight line, one subject after the other; 2) *parallel*, identifying a movement along parallel lines; 3) *circular*, corresponding to the two subjects following parallel and circular trajectories; 4) *free*, where all subjects could move freely in the room; 5) *paral-diag*, identifying a movement on parallel lines but with the subjects spaced apart along the movement directions; and 6) *vs-in-line*, where the two subjects moved one toward the other following the same linear trajectory. All trajectories are depicted in Fig. 3, while Fig. 4 shows a picture of the implemented experimental setup. In total, we collected 55 sequences, each 40 s long. In every sequence, subjects were tracked by all four radars simultaneously and independently.

Then, tracking information has been fused considering all possible combinations of 1–4 radars. However, we experienced the Jetson Nano DevKit edge computer not being able to properly track more than two targets simultaneously, and we noticed the issue after the experiments. For this reason, to provide reliable results, we decided to show results with up to three fused radars. After filtering out the corrupted data, considering all the evaluated combinations, we analyzed a total of 220, 187, and 91 experiments for 1, 2, and 3 fused radars, respectively.

Moreover, to assess the performance of ORACLE in cases when furniture and obstacles are present in the environment, we conducted additional experiments recreating a common office environment, with a table, chairs, and a hanger, and simulating the corner of a corridor. For this part of the experiment, we collected 40 sequences with two subjects and three radars. Results pertaining to this part of the dataset will be presented in Section V-G1.

C. Evaluation Metrics

To evaluate the self-calibration algorithm performance, we define the *orientation error* as the absolute value of the difference between the true orientation angle of a radar and the estimated one. This is derived from the corresponding rotation matrix, after calibration, as explained in Section III-C. The *position error* is defined as the Euclidean distance between the estimated position of the radar and its true position. To assess the tracking performance, we adopt the *multiple object tracking performance accuracy* (MOTA) metrics, which accounts for the number of misses, false positives, and switches in the object detections, and the *multiple object tracking performance precision* (MOTP) metric, which represents the mean position error by considering only correctly tracked objects. More details about these metrics can be found in [37].

D. Self-Calibration

As previously mentioned, in the first part of this work, we are presenting an enhanced version of mmSCALE [13],
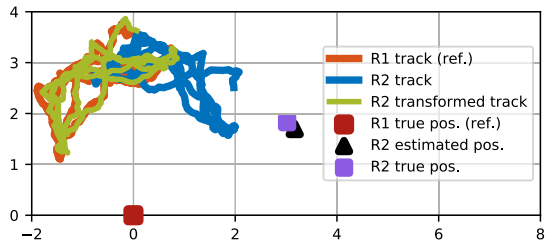
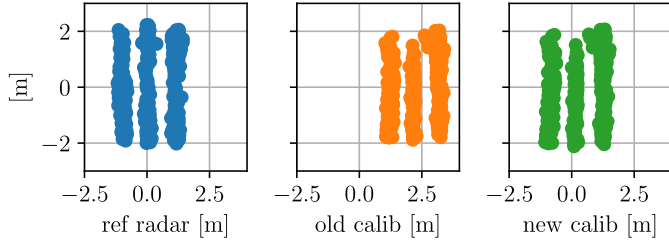Fig. 5. Example of self-calibration with a *free* trajectory.

Fig. 6. Example of a situation where the old algorithm fails while the new one does not. In this case, linear, similar trajectories led to a mistake in the track association phase, resulting in a shift in the estimated location of the radar.
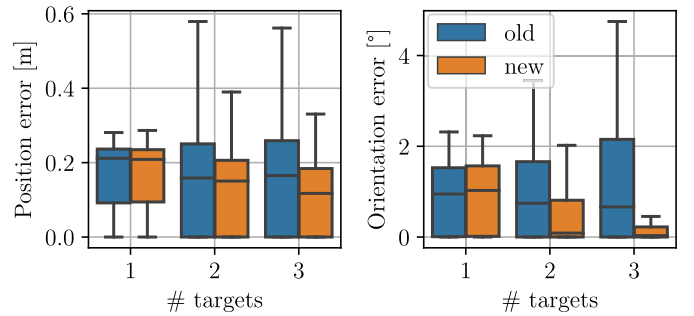

Fig. 7. Comparison of the self-calibration results when using the old and the new self-calibration algorithm as a function of the number of targets tracked during the calibration phase.

TABLE I
COMPARISON OF SELF-CALIBRATION ALGORITHMS

	Median Error (IQR)	Old	New	New-Old Diff.
1T	**Position [m]**	0.21(0.14)	0.21(0.14)	±0.00(±0.00)
	Orientation [°]	0.95(1.50)	1.00(1.60)	+0.05(−0.10)
2T	**Position [m]**	0.16(0.25)	0.15(0.21)	−0.01(−0.04)
	Orientation [°]	0.75(1.70)	0.09(0.81)	−0.66(−0.89)
3T	**Position [m]**	0.17(0.26)	0.12(0.18)	−0.05(−0.08)
	Orientation [°]	0.66(2.20)	0.03(0.22)	−0.63(−1.98)

our self-calibration algorithm. The enhancement, consisting in the addition of the *masking* phase to the self-calibration procedure, allows for handling a wider range of cases, where the previous version was more prone to errors, and to achieve a general improvement in the accuracy of the calibration parameters estimation. To show the effect of the enhancement, we will focus on the comparison between the old and the new version of the self-calibration algorithm.

1) Qualitative Results: Fig. 5 shows a qualitative example of the calibration process. Here, after finding the optimal rotation and translation parameters, we applied the rigid transformation to the trajectory seen by radar 2 (blue line, R2), to superimpose it with the one of radar 1 (orange line, R1). The transformed trajectory (green line) matches the reference one well, showing a good calibration result. We represent the reference radar with a red square (located at $[0, 0]^T$), while the black triangle and the purple square mark the estimated position of radar 2 and its GT, respectively.

As long as only one target is being tracked, the track association phase is easy and it is likely that no errors occur. If the number of tracked targets increases, the track association phase becomes more challenging. Our track association cost (see Section IV-C2) can lead to a correct association for most situations. However, there are some particular cases where it is not sufficient. Fig. 6 shows an example of one of such cases with three subjects following a *parallel* trajectory. The blue lines represent the trajectories of the reference radar in its RS, the orange lines show the trajectories of another radar after transforming them in the reference radar's RF using the old self-calibration algorithm, while the green lines represent the trajectories from the same radar after transforming them using the new self-calibration algorithm. In this scenario, all trajectories are very similar, as the subjects proceed in parallel and at the same speed. Because of the very high correlation between the track positions over time, the association costs are very similar and the final association result depends on subtle

numerical variations due to the tiny differences between the trajectory shapes. From the figure, we notice that this causes a wrong association between the tracks from the reference radar and the other one, reflecting in a shifted estimate of the radar's RS, when using the old method. The new method, instead, correctly copes with this situation.

2) Position and Orientation Errors: Fig. 7 shows a comparison between the calibration performance using the old and the new version of the self-calibration algorithm versus the number of targets tracked. When only one target is tracked, there is almost no difference between the two algorithms, while a clear improvement is observed as the number of targets increases. In particular, the new algorithm has a great effect in reducing the sparsity of the box plots, meaning it is increasing the number of cases it can handle correctly. Table I shows the numerical results in terms of median and interquartile range (IQR). It also shows the difference between the new and the old algorithm.

E. FC Tracking Accuracy

We evaluate the performance of the fusion algorithm in two cases: 1) using the roto-translation parameters obtained from the GT and 2) using their estimations from the self-calibration algorithm. To evaluate ORACLE in a more realistic scenario, when using self-calibration, the transformation parameters are computed only once per setup-trajectory pair, that is, the first time a sequence with that setup-trajectory pair is elaborated. Then, all sequences with the same setup and trajectory type use the same parameters. Fig. 8 shows the tracking results versus the number of targets tracked and the number or radars used for the fusion. In the figure, xT-SF and xT-GT denote a case where x targets are tracked and, respectively, self-calibration or GT are used. The bar charts represent the 1 standard deviation for the self-calibration results. The values are computed as the average over all the sequences
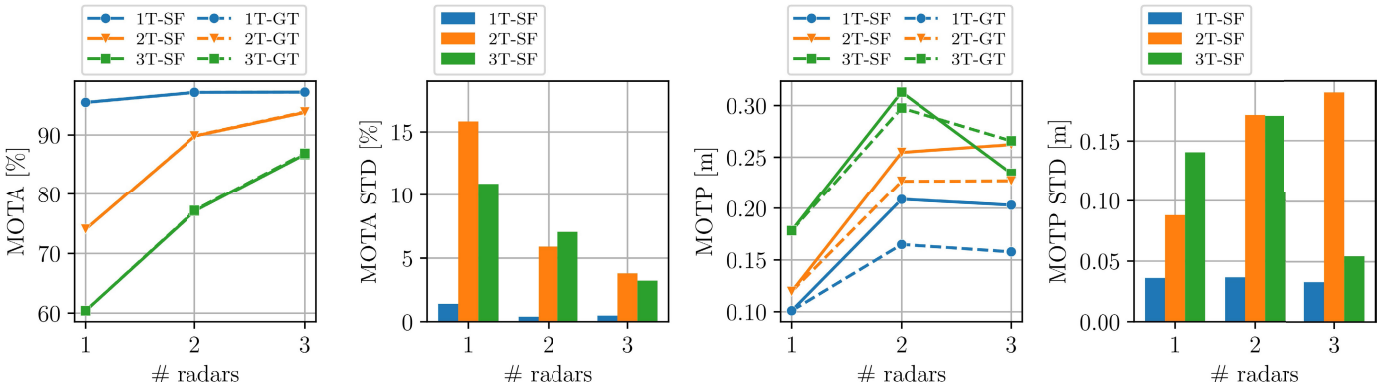
Fig. 8. Average MOTA and MOTP as a function of the number of radars used for the fusion and of the number of targets tracked. Solid and dashed lines identify, respectively, results obtained using self-calibration (SF) and GT to estimate radars' location and orientation. The bar charts represent the 1 standard deviation for the SF case.

TABLE II
SUMMARY OF ORACLE'S TRACKING PERFORMANCE

		1R	2R	3R	3R-1R diff.
1T	MOTA [%]	95 ± 2	97 ± 0	97 ± 0	$+2$
	MOTP [m]	0.10 ± 0.04	0.21 ± 0.04	0.20 ± 0.03	$+0.10$
2T	MOTA [%]	74 ± 13	90 ± 6	94 ± 4	$+20$
	MOTP [m]	0.12 ± 0.09	0.25 ± 0.17	0.26 ± 0.19	$+0.14$
3T	MOTA [%]	60 ± 11	77 ± 7	87 ± 3	$+27$
	MOTP [m]	0.18 ± 0.14	0.31 ± 0.17	0.23 ± 0.05	$+0.05$

with the same number of radars and targets. Numerical results using self-calibration are presented in Table II, where xR is used to indicate that x radars were used for the fusion.

When only one target is tracked, there is almost no difference between using a single radar or multiple fused radars ($+2\%$). As the number of tracked people increases, single sensors experience a remarkable decrease in the MOTA (-35%) while the FC maintains high performance, with a MOTA as high as 87% when three targets are tracked, leading to an improvement with respect to single sensors of $+27\%$. This is because multiple targets may often create occlusions with respect to single radars, increasing the number of misses and switches in the tracks. Instead, occlusions can be mitigated by fusing data from different points of view. We also note that GT and self-calibration achieve very similar results in terms of MOTA.

In general, MOTP slightly increases in the fused tracking. The reason is twofold. First, noise can be incorporated during the fusion process and slightly affect the localization performance. Second, MOTP is computed only for correctly tracked targets. Because a single radar's tracking capability is limited, successful tracking occurs only in sufficiently simple scenarios, where MOTP would be straightforwardly low. On the contrary, multiple radars track targets successfully even in more complicated cases. This increases the range of points for which we compute MOTP to include more arduous and inevitably less precise location and movement estimates. Interestingly, for the 1T and 2T cases, after increasing from 1 to 2 radars, MOTP is almost constant when moving from 2 to 3 fused radars, suggesting that this could be the case also if more radars are fused. Following the 3T lines, instead, MOTP increases from 1 to 2 radars and then decreases when three radars are used, reaching the same values

of the 2T lines. A possible explanation for this is that two radars are not enough to track three targets, leading to errors in the track associations that cause the MOTP to increase. Three radars, instead, have better tracking capabilities and can better handle three targets. For the most challenging scenario (three targets), MOTP is 31 cm when two radars are used and 23 cm when three radars are used. MOTP is slightly lower when using GT rather than self-calibration because of the more precise knowledge about the sensors' position and orientation.

As a final test, we acquired some sequences where we fuse all of the four radars for various two-target trajectories, providing a MOTA and MOTP of 95% and 20 cm, respectively, while single radars on the same sequences reach a MOTA and MOTP of 78% and 11 cm, respectively. Since, for this test, we only collected a few sequences that do not represent a statistically significant set, we present them only as an example.

These results show that our self-calibration algorithm works well in combination with the proposed fusion algorithm and that they can be effectively used together to enable occlusion-resilient people tracking through a self-calibrated radar network, requiring almost no human intervention.

F. Robustness to Reduced Fusion Rate

In certain resource-constrained applications, it may be useful to reduce the FC processing rate of the sensor data, to lower the computational burden. However, this requires striking a balance between fusion rate and tracking accuracy, as decreasing the processing rate reduces the capability of the FC to follow the movement of the subjects.

In Fig. 9, we show the MOTA and MOTP curves as a function of the ratio T_c/T_s. This is varied by fixing $T_s = 66.7$ ms (15 fps) and changing T_c from $0.2T_s$ to $25T_s$. Values are obtained by averaging all experiments with three radars. We can identify three regions.

1) For $T_c/T_s < 0.8$, the MOTA is very low, as the FC runs significantly faster than the sensors and, therefore, has to rely mostly on the KF predictions, which are inaccurate after a few consecutive steps. The MOTP instead is unaffected as it is obtained only on the successfully tracked subjects.
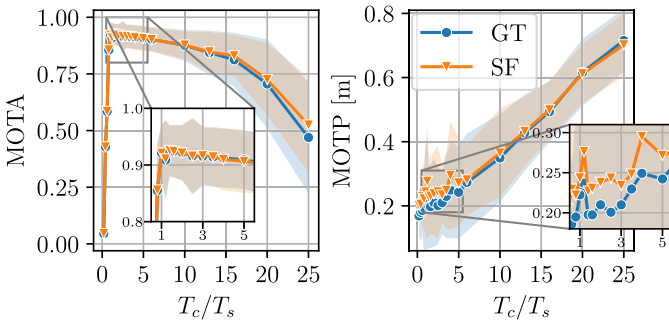
Fig. 9. Average MOTA and MOTP versus different values of the ratio T_c/T_s when calibration is performed using GT measurements (GT), or with the self-calibration (SF). The shaded areas represent 1 standard deviation.

2) For $0.8 \leq T_c/T_s \leq 5$, our system achieves the best performance in terms of MOTA, that is, over 90%. At the same time, the MOTP is still low, with errors of less than 29.4 cm when using self-calibration with $T_c/T_s = 5$. This shows that, if necessary, the processing load on the FC can be *reduced by* five *times* with negligible performance degradation.

3) For $T_c/T_s > 5$, MOTA degrades slowly and MOTP increases. This is because the time step of the FC, especially toward the end of this region, is too long to accurately follow human movement using the CV model.

Finally, we notice that the MOTA is almost unaffected by using self-calibration in place of the GT sensor locations and orientations. This holds for all regions 1)–3). However, as expected, the MOTP is slightly worse in case self-calibration is used, as the residual error in the locations of the sensors indirectly affects the FC tracking precision.

G. Impact of Obstacles and Furniture

In this section, we evaluate the tracking performance of ORACLE in two challenging scenarios characterized by the presence of furniture and obstacles, as represented in Fig. 10.

1) Presence of Furniture: The first scenario contains furniture, placed as shown in Fig. 10(b). Subjects walk in a typical office setting including a table, chairs, and a hanger, following linear ad-free trajectories. The MOTA and MOTP obtained by ORACLE in this case are reported in Table III, with one (1T) and two (2T) targets moving in the room. By comparing the obtained results with those in Table II referring to the same number of radars and targets (but without any furniture), we observe that the furniture has an impact on the performance with one radar (1R), decreasing the MOTA of −20% in the Furn-1T case and of −9% in Furn-2T, respectively. Instead, when data from three radars (3R) are fused, the MOTA is unaffected, proving that ORACLE mitigates the effect of furniture on the tracking performance. We remark that, in this setup, we purposely placed the radars at the same height as the table and the chairs, to create a challenging scenario where part of the radio reflections are occluded by the objects. This prevents the correct tracking of the subjects when a single radar is used and occlusions occur, whereas the tracking performance is unaffected when multiple radars combine their data via the proposed data fusion algorithm.

2) U-Shaped Corner: In the second scenario, which we refer to as a "U-shaped corner," we place a large, wall-like obstacle in the middle of the room. The obstacle is made with metal whiteboards covered with foam rubber, which completely blocks the mmWave signal. This replicates a U-shaped corner in which targets are entirely occluded for some of the radar sensors [see Fig. 10(c)], while they are always in the FoV of at least one sensor. Specifically, targets moving in the proximity of radar 1 in Fig. 10(c) cannot be detected by radar 2 and vice versa. Table III shows the MOTA and MOTP metrics obtained by ORACLE in this scenario, compared against the results for a single radar sensor (radar 2). The subject was instructed to move according to free and linear trajectories, and the reported results show the combined resulting metrics. MOTA and MOTP performance obtained by ORACLE show that our system is not affected by the presence of a large obstacle. This is a result of the combined view provided by the FC, which is robust to the subjects being undetectable by a subset of the available radars. Conversely, the MOTA of the single radars is reduced on average, as a result of the subjects being occluded by the U-shaped corner for a fraction of the measurement time.

As a final remark, we underline that the performance of ORACLE in the U-shaped corner scenario also serves as an experimental evaluation of the robustness of its track initialization and termination procedure, described in Section IV-D4. It follows that such a procedure is robust to subjects that frequently enter and leave the FoV of radar sensors due to occlusions.

H. Impact of Tracking on Localization Error

Our experimental results show that applying a tracking algorithm to smooth the raw radar measurements and handle their association with the different subjects is crucial. This is especially true for multiple targets scenarios, where there is ambiguity regarding which target generates which reflection. Fig. 11 shows the empirical localization error cdf of the single sensors using tracking, compared to that obtained from the raw measurements, for 2 and 3 moving targets. There is a clear gain in applying our tracking mechanism, which avoids the error have a long-tailed distribution. Raw measurements are unreliable in multipath-rich indoor environments, as they are subject to fluctuations, as well as missed and ghost detections. Note that the use of a (multisensor) tracking step sets ORACLE apart from other approaches, such as [12]: while these previous works apply measurement-level centralized data fusion without tracking, relying on raw radar measurements, ORACLE performs track-level fusion. ORACLE achieves better localization performance, while significantly lowering network overhead, as only the low-dimensional tracks need to be communicated to the FC.

I. Impact of Radars' Location on Fused Tracking

When tackling the problem of people tracking through multiple radars, it is interesting to explore how different radar deployments affect the results. This kind of study is beyond the scope of this article and would require a denser deployment
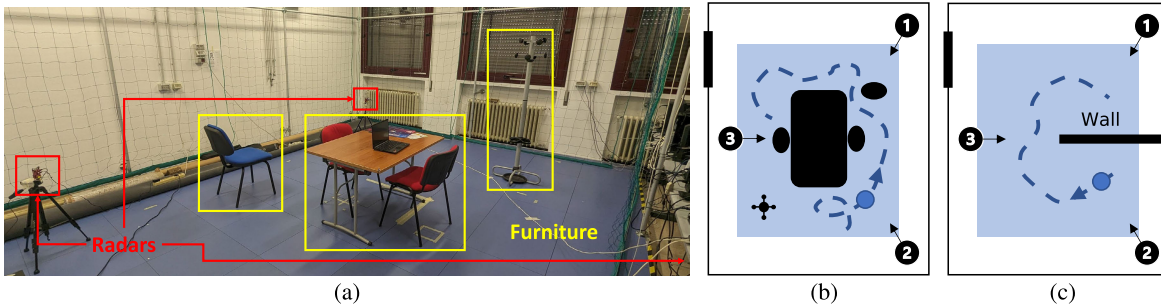
Fig. 10. To the left, (a) shows the setup with furniture. To the right, we represent the (b) schemes of the setups with furniture and (c) simulating the U-shaped corner of a corridor.

TABLE III
TRACKING PERFORMANCE WITH FURNITURE (FURN-) AND IN A U-SHAPED CORNER

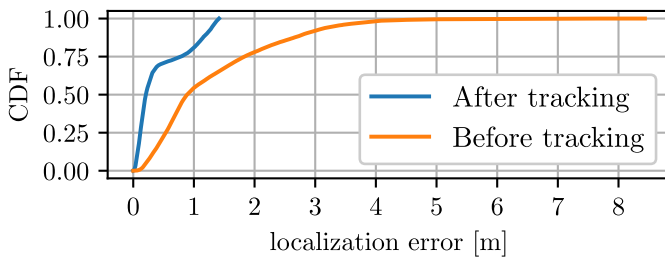|		1R	3R	3R-1R diff.
Furn-1T	**MOTA [%]**	75 ± 8	96 ± 2	$+21$
	MOTP [m]	0.19 ± 0.02	0.16 ± 0.01	-0.03
Furn-2T	**MOTA [%]**	65 ± 10	94 ± 3	$+29$
	MOTP [m]	0.16 ± 0.03	0.16 ± 0.01	± 0
U-shaped corner	**MOTA [%]**	50 ± 7	94 ± 4	$+44$
	MOTP [m]	0.17 ± 0.02	0.18 ± 0.03	$+0.01$

Fig. 11. Empirical localization error cdf with and without using a tracking algorithm on the single sensors, thus relying only on the raw radar measurements.
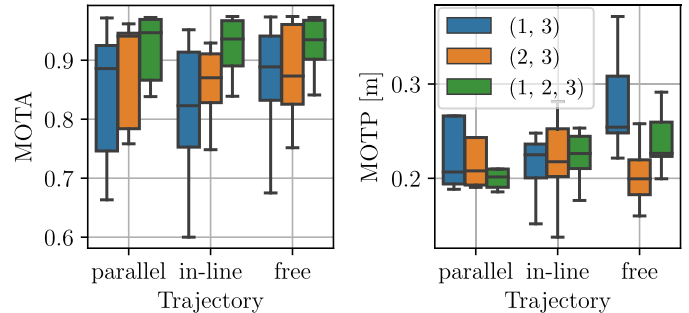

Fig. 12. Box plots of MOTA and MOTP for particular trajectories, as a function of specific combinations of fused radar. More precisely, (1,3) has facing radars, (2, 3) has perpendicular radars, (1, 2, 3) fuses all of the three. Radar numbers correspond to those in Fig. 3, first row (setup-1).

of radars. For this reason, here we show only some preliminary results, while we leave a deeper inspection of the problem to future developments. In Fig. 12, we compare the results for three trajectories and some specific combinations of radars for the fusion. In particular, according to setup-1 (see Fig. 3, first row), combination $(1, 3)$ corresponds to two radars facing each other, combination $(2, 3)$ has perpendicular radars, while combination $(1, 2, 3)$ fuses all of the three. For each combination, results are computed using self-calibration and averaging over all sequences featuring the particular trajectory, with either 1, 2, or 3 targets. Considering MOTA, perpendicular radars are generally better than facing radars, while fusing three radars always provides the best results. Regarding MOTP, there is no clear trend common to all trajectories. The median MOTP is always within the $[0.20, 0.25]$ m interval, which means that there are no great differences depending on the combinations. The only case worth mentioning is that featuring free trajectory and $(1, 3)$ combination, where MOTP values are generally higher than in the other cases. In conclusion, from this brief analysis, it appears that: 1) a larger number of radars is to be preferred over a lower one and 2) radars with more diverse points of view provide, in general, better tracking results.

J. Comparison With the MHT Algorithm

ORACLE radar network calibration and fusion techniques require the edge sensors to only provide a state vector and a covariance matrix for each target, along with the corresponding timestamp. This design provides flexibility in the choice of the tracking algorithm for the radar devices, as any can be implemented, as long as it returns the desired state vector and covariance information. To demonstrate ORACLE's flexibility, we replaced the NN-JPDAF tracking at the edge computers (described in Section III-B) with the well-known multiple hypothesis tracking (MHT) scheme [38]. For MHT, we used the same inputs as for NN-JPDAF, that is, we applied DBSCAN [30] clustering to the radar point clouds and used the clusters centroids as detections. MHT was tested on a subset of the dataset including two targets and *free* and *parallel* trajectories, presenting the results in Table IV. Both NN-JPDAF and MHT show an improvement as the number of fused radar tracks increases (from one, 1R, to three, 3R), with NN-JPDAF performing, overall, slightly better than MHT.

K. Summary of the Implementation Challenges

In addition to the challenges posed by ORACLE's design, presented in Section III, its experimental implementation and validation involve several other important aspects. The main ones are discussed in the following.

1) Communication Between Network Nodes: ORACLE's deployment requires enabling communication between network nodes. In particular, it is necessary to have a management unit that allows controlling the beginning and the end of the experimental sequences and that allows exchanging data

TABLE IV
COMPARISON WITH THE MHT ALGORITHM

		1R	2R	3R
ORACLE	**MOTA [%]**	83 ± 6	91 ± 6	96 ± 1
	MOTP [m]	0.08 ± 0.02	0.27 ± 0.19	0.2 ± 0.01
ORACLE with MHT	**MOTA [%]**	76 ± 22	83 ± 15	90 ± 6
	MOTP [m]	0.32 ± 0.17	0.2 ± 0.03	0.19 ± 0.02

between the edge nodes and the FC. In ORACLE, this is solved by using the RadNet platform [14], which provides all the necessary functionalities.

2) Time Synchronization Between Network Nodes: As explained in Section IV, ORACLE relies on a loose time synchronization between the devices (i.e., not at clock level). This is achieved by using standard NTP that allows synchronizing the local time of the edge computers to the millisecond level. This is important because the radar measurements are timestamped with the device's time after being acquired, and the timestamp is then used for the calibration and fusion processes. However, as mentioned in Section V-B, we experienced the Jetson Nano DevKit edge computer not being able to timely track more than two targets simultaneously, due to its low computational power, introducing a delay in the data timestamps and, therefore, making them unusable.

VI. CONCLUSION

In this work, we presented ORACLE, a solution to the mmWave radar network deployment and integration problem for human sensing purposes. First, ORACLE automatically estimates the relative position and orientation of the radars with respect to a common RS. Then, it exploits such estimates to fuse the information about people tracked by different radars at an FC, enhancing the resilience of the subject localization in case of occlusions. ORACLE estimates the radars' position and orientation with a median error of 0.12 m and 0.03°, respectively, exploiting the movement trajectories of tracked people. With respect to existing self-calibration techniques, ORACLE is more robust to multiple subjects concurrently moving in the environment, with no need to follow any predetermined trajectory for the calibration. By fusing multiple radars tracking information, ORACLE improves on single sensors by up to 27% in mean tracking accuracy, with a mean precision of 23 cm in the most challenging case of three targets moving. ORACLE handles different time steps for the single sensors and the FC, keeping the tracking accuracy higher than 90% when the ratio between the central and the sensors time step is $0.8 \leq T_c/T_s \leq 5$. Finally, when evaluated in challenging indoor scenarios, ORACLE shows an improvement in the tracking accuracy by up to 29% when furniture is present in the room and demonstrates its capability to effectively handle situations where tracking is required around corridor corners. These results substantiate ORACLE as a key technology enabler for distributed people tracking with radar networks, serving as a base system for a large variety of applications, from personnel recognition to restricted areas monitoring, elderly care, customer profiling, and many others.

Future research includes the extension of ORACLE to multiple disjoint radar networks (where none of the radars of a network share a part of the FoV with any of the radars of the other networks), and the study of how different points of view of the same scene affect the tracking performance.

ACKNOWLEDGMENT

REFERENCES

[1] S. A. Shah and F. Fioranelli, "RF sensing technologies for assisted daily living in healthcare: A comprehensive review," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 34, no. 11, pp. 26–44, Nov. 2019.

[2] S. Savazzi, S. Sigg, F. Vicentini, S. Kianoush, and R. Findling, "On the use of stray wireless signals for sensing: A look beyond 5G for the next generation of industry," *Computer*, vol. 52, no. 7, pp. 25–36, Jul. 2019.

[3] A. Shastri et al., "A review of millimeter wave device-based localization and device-free sensing technologies and applications," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 3, pp. 1708–1749, 3rd Quart., 2022.

[4] J. Pegoraro and M. Rossi, "Real-time people tracking and identification from sparse mm-wave radar point-clouds," *IEEE Access*, vol. 9, pp. 78504–78520, 2021.

[5] P. Zhao et al., "MID: Tracking and identifying people with millimeter wave radar," in *Proc. 15th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, May 2019, pp. 33–40.

[6] M. Canil, J. Pegoraro, and M. Rossi, "MilliTRACE-IR: Contact tracing and temperature screening via mmWave and infrared sensing," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 2, pp. 208–223, Feb. 2022.

[7] A. D. Singh, S. S. Sandha, L. Garcia, and M. Srivastava, "RadHAR: Human activity recognition from point clouds generated through a millimeter-wave radar," in *Proc. 3rd ACM Workshop Millimeter-Wave Netw. Sens. Syst.*, Oct. 2019, pp. 51–56.

[8] M. Shen, K.-L. Tsui, M. A. Nussbaum, S. Kim, and F. Lure, "An indoor fall monitoring system: Robust, multistatic radar sensing and explainable, feature-resonated deep neural network," *IEEE J. Biomed. Health Informat.*, vol. 27, no. 4, pp. 1891–1902, Apr. 2023.

[9] M. J. Bocus and R. J. Piechocki, "Passive unsupervised localization and tracking using a multi-static UWB radar network," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 1–6.

[10] S. Li, J. Guo, R. Xi, C. Duan, Z. Zhai, and Y. He, "Pedestrian trajectory based calibration for multi-radar network," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, May 2021, pp. 1–2.

[11] S. Iwata, T. Koda, and T. Sakamoto, "Multiradar data fusion for respiratory measurement of multiple people," *IEEE Sensors J.*, vol. 21, no. 22, pp. 25870–25879, Nov. 2021.

[12] Z. Xie and F. Ye, "Self-calibrating indoor trajectory tracking system using distributed monostatic radars for large scale deployment," in *Proc. 9th ACM Int. Conf. Syst. Energy-Efficient Buildings, Cities, Transp.*, Boston, MA, USA, Nov. 2022, pp. 188–197.

[13] A. Shastri, M. Canil, J. Pegoraro, P. Casari, and M. Rossi, "MmSCALE: Self-calibration of mmWave radar networks from human movement trajectories," in *Proc. IEEE Radar Conf. (RadarConf)*, New York, NY, USA, Mar. 2022, pp. 1–6.

[14] E. Bashirov, M. Canil, and M. Rossi, "RadNet: A testbed for mmWave radar networks," in *Proc. 1st Int. Workshop Emerg. Topics Wireless*, Rome, Italy, Dec. 2022, pp. 1–5.

[15] V. S. Chernyak, *Fundamentals of Multisite Radar Systems: Multistatic Radars and Multistatic Radar Systems*. Boca Raton, FL, USA: CRC Press, 1998.

[16] F. Hoffmann, M. Ritchie, F. Fioranelli, A. Charlish, and H. Griffiths, "Micro-Doppler based detection and tracking of UAVs with multistatic radar," in *Proc. IEEE Radar Conf. (RadarConf)*, Philadelphia, PA, USA, May 2016, pp. 1–6.

[17] A. Ledergerber and R. D'Andrea, "A multi-static radar network with ultra-wideband radio-equipped devices," *Sensors*, vol. 20, no. 6, p. 1599, Mar. 2020.

[18] S. Bartoletti, A. Conti, A. Giorgetti, and M. Z. Win, "Sensor radar networks for indoor tracking," *IEEE Wireless Commun. Lett.*, vol. 3, no. 2, pp. 157–160, Apr. 2014.

[19] Y. He, F. Le Chevalier, and A. G. Yarovoy, "Range-Doppler processing for indoor human tracking by multistatic ultra-wideband radar," in *Proc. 13th Int. Radar Symp.*, Warsaw, Poland, May 2012, pp. 250–253.

[20] H. Shu and Q. Liang, "Data fusion in a multi-target radar sensor network," in *Proc. IEEE Radio Wireless Symp.*, Long Beach, CA, USA, Jan. 2007, pp. 129–132.

[21] F. Folster and H. Rohling, "Data association and tracking for automotive radar networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 4, pp. 370–377, Dec. 2005.

[22] M. Rykunov, E. De Greef, H.-U.-R. Khalid, K. Aziz, A. Bourdoux, and H. Sahli, "Multi-radar fusion for failure-tolerant vulnerable road users classification," in *Proc. 18th Eur. Radar Conf. (EuRAD)*, Apr. 2022, pp. 337–340.

[23] A. Ahmed, S. Zhang, and Y. D. Zhang, "Multi-target motion parameter estimation exploiting collaborative UAV network," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 4459–4463.

[24] R. G. Guendel, M. Unterhorst, E. Gambi, F. Fioranelli, and A. Yarovoy, "Continuous human activity recognition for arbitrary directions with distributed radars," in *Proc. IEEE Radar Conf. (RadarConf)*, May 2021, pp. 1–6.

[25] Z. Meng et al., "Gait recognition for co-existing multiple people using millimeter wave sensing," in *Proc. Conf. Artif. Intell. (AAAI)*, Feb. 2020, pp. 849–856.

[26] E. I. P. Copa, K. Aziz, M. Rykunov, E. De Greef, A. Bourdoux, and F. Horlin, "Radar fusion for multipath mitigation in indoor environments," in *Proc. IEEE Radar Conf. (RadarConf)*. Florence, Italy: IEEE, Sep. 2020, pp. 1–5.

[27] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Process. Mag.*, vol. 34, no. 2, pp. 22–35, Mar. 2017.

[28] T. Wagner, R. Feger, and A. Stelzer, "Radar signal processing for jointly estimating tracks and micro-Doppler signatures," *IEEE Access*, vol. 5, pp. 1220–1238, 2017.

[29] C.-Y. Chong, S. Mori, W. H. Barker, and K.-C. Chang, "Architectures and algorithms for track association and fusion," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 15, no. 1, pp. 5–13, Jan. 2000.

[30] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. ACM Conf. Knowl. Discovery Data Mining (KDD)*, Aug. 1996, pp. 1–6.

[31] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.

[32] Y. Bar-Shalom, F. Daum, and J. Huang, "The probabilistic data association filter," *IEEE Control Syst. Mag.*, vol. 29, no. 6, pp. 82–100, Dec. 2009.

[33] O. Sorkine-Hornung and M. Rabinovich, "Least-squares rigid motion using SVD," *Computing*, vol. 1, no. 1, pp. 1–5, Jan. 2017.

[34] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics*, vol. 52, no. 1, pp. 7–21, Feb. 2005.

[35] K. C. Chang, R. K. Saha, and Y. Bar-Shalom, "On optimal track-to-track fusion," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 33, no. 4, pp. 1271–1276, Oct. 1997.

[36] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 42, no. 1, pp. 80–86, Feb. 2000.

[37] K. Bernardin, A. Elbs, and R. Stiefelhagen, "Multiple object tracking performance metrics and evaluation in a smart room environment," in *Proc. 6th IEEE Int. Workshop Vis. Surveill.*, May 2006, vol. 90, no. 91, pp. 1–8.

[38] D. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Autom. Control*, vol. AC-24, no. 6, pp. 843–854, Dec. 1979.

Marco Canil (Graduate Student Member, IEEE) received the M.Sc. degree in ICT for internet and multimedia engineering from the Department of Information Engineering (DEI), University of Padua (UNIPD), Padua, Italy, in 2020, where he is currently pursuing the Ph.D. degree in information engineering.

In 2023, he was a Visiting Research Scholar at the New York University Tandon School of Engineering, New York, NY, USA. His research interests include machine learning, signal processing, and sensor fusion for sensing with millimeter-wave systems.

Jacopo Pegoraro (Member, IEEE) received the Ph.D. degree in information engineering from the University of Padua, Padua, Italy, in 2023.

He is currently working as a Postdoctoral Researcher at the Department of Information Engineering, University of Padua. He was a Visiting Research Scholar at the New York University Tandon School of Engineering, New York, NY, USA, in 2022. His research interests include signal processing and machine learning for millimeter-wave sensing and integrated sensing and communication.

Dr. Pegoraro received the Best Original Work Award at the 2023 IEEE SPS/AESS and EURASIP Summer School on Integrated Sensing and Communication.

Anish Shastri (Graduate Student Member, IEEE) received the B.Tech. degree in electronics and communication engineering (ECE) from the National Institute of Technology (MANIT), Bhopal, India, in 2016, and the M.S. degree (by Research) in ECE from the International Institute of Information Technology – Hyderabad (IIIT-H), Hyderabad, India, in 2019. He is currently pursuing the Ph.D. degree with the Department of Information Engineering and Computer Science, University of Trento, Povo, Italy.

He is an Early Stage Researcher with the EU H2020 Marie Skłodowska Curie Actions MINTS ETN. Before starting the Ph.D. degree, he was a Research Intern at the Indoor Networks Research Group, Nokia Bell Labs, Dublin, Ireland. His research interests include signal processing and machine learning for wireless communications, with his current research focusing on algorithms for mmWave indoor localization, environment mapping, and sensing.

Paolo Casari (Senior Member, IEEE) received the Ph.D. degree in information engineering from the University of Padua, Padua, Italy, in 2008.

From 2015 to 2019, he was with IMDEA Networks, Madrid, Spain, where he led the Ubiquitous Wireless Networks Group. He is an Associate Professor at the Department of Information Engineering and Computer Science, University of Trento, Povo, Italy. He is currently the PI of the NATO SPS Project SAFE-UComm. His research interests include many aspects of networked communications, such as channel modeling, network protocol design, localization, simulation, and experimental evaluations.

Prof. Casari received two Best Paper Awards. He regularly collaborates with the organizing committee of several international conferences. He serves on the Editorial Board of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He was an Associate Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING from 2018 to 2023 and is currently a Co-Guest Editor of two special issues for Elsevier's *Computer Communications* and *Computer Networks* journals.

Michele Rossi (Senior Member, IEEE) is a Professor of Wireless Networks at the Department of Information Engineering (DEI), University of Padua (UNIPD), Padua, Italy, where he is the Head of the Master's degree in ICT for Internet and Multimedia. He also teaches Human Data Analysis at the Data Science Master's degree at the Department of Mathematics (DM), UNIPD. Since 2017, he has been the Director of the DEI/IEEE Summer School of Information Engineering. Over the years, he has been involved in several EU projects on wireless sensing and Internet of Things (IoT) and has collaborated with major companies such as Ericsson, DOCOMO, Samsung, and INTEL. His research interests lie broadly in wireless sensing systems, green mobile networks, edge and wearable computing.

Dr. Rossi was a recipient of seven Best Paper Awards from IEEE and currently serves on the Editorial Boards of IEEE TRANSACTIONS ON MOBILE COMPUTING and the Open Journal of the Communications Society. His research is currently supported by the European Commission through the H2020 Projects MINTS under Grant 861222 on "mmWave Networking and Sensing" and GREENEDGE under Grant 953775 on "Green Edge Computing for Mobile Networks" (Project Coordinator).