UNIVERSITY OF PADOVA

DEPARTMENT OF INFORMATION ENGINEERING

Ph.D. Course in Information Engineering

Information and Communication Science and Technologies Curriculum

XXXVI series

Collaborative Human Sensing with mmWave Systems

Ph.D. Candidate
Marco Canil

Ph.D. Supervisor
Professor Michele Rossi

Ph.D. Coordinator
Professor Fabio Vandin

Academic Year
2022–2023

Abstract

Remote perception of human movements has the potential to revolutionize the way we interact with technology, enabling an unprecedented integration in everybody's daily life. In this panorama, radio detection and ranging (RADAR) devices working in the millimeter-wave (mmWave) region of the radio spectrum have sparked great interest from academia to industry, as they combine highly accurate sensing capabilities with appealing properties of mmWaves, such as insensitivity to extreme light conditions and to the presence of dust, smoke, or rain. Moreover, mmWave RADARs raise less privacy concerns than vision-based monitoring systems, as no image of the surroundings is captured.

However, commercial mmWave radar devices have limited range (up to 6–8 m) and are subject to occlusion, which may constitute a significant drawback in large, crowded rooms containing furniture and walls. Thus, covering large indoor spaces requires multiple RADARs, with *known* relative position and orientation and algorithms to combine their output information.

In this thesis, we focus on providing practical solutions for the adoption of mmWave RADARs in real-world settings. In particular, we devise algorithms for the automatic deployment of RADAR networks and for their use for human sensing.

Initially, we develop a method for the *self-calibration* of RADAR sensor networks. The problem is to automatically estimate the relative position and orientation of the RADARs to enable data fusion. The proposed solution works by leveraging the trajectories of people moving freely in the common field of view (FoV) of the RADARs, requiring no human intervention. Then, we develop an experimental testbed for the easy deployment and testing of RADAR network algorithms.

Subsequently, we tackle the problem of data fusion in the context of mmWave monitoring systems. We address this in three ways: *(i)* considering a system where multiple RADARs' data are fused to provide a unique and unified people tracking, *(ii)* contemplating the cooperation of mmWave RADARs with other sensors, and *(iii)*, exploiting communication devices for sensing purposes. In *(i)*, each node of the RADAR network is endowed with resource-constrained computational capabilities, performs people tracking independently, and shares its tracking information with a *fusion center* that fuses data providing an enhanced, unified tracking among all sensors. In *(ii)*, a thermal camera (TC) is used in conjunction with a mmWave RADAR to provide concurrent contact tracing and body temperature screening. Finally, in *(iii)*, the use of communication devices is considered in an integrated sensing and communication (ISAC) scenario. In the latter, human sensing parameters are extracted from the communication packets exchanged between one transmitter (TX) and multiple receivers (RXs).

Contents

Listing of figures

Listing of tables

Listing of acronyms

μD	micro-Doppler
AoA	angle of arrival
AoD	angle of departure
BP	beam pattern
BW	bandwidth
CFAR	constant false alarm rate
CFO	carrier frequency offset
CIR	channel impulse response
CV	constant-velocity
DBSCAN	density-based spatial clustering of applications with noise
DFT	discrete Fourier transform
EFK	extended Kalman filter
EM	electromagnetic
FC	fusion center
FMCW	frequency-modulated continuous wave
FoV	field of view
GM	Gaussian mixture
GT	ground-truth
IP	Internet Protocol
IQR	interquartile range
ISAC	integrated sensing and communication

KF	Kalman filter
LO	local oscillator
LoS	line-of-sight
LS	least squares
MHT	Multiple Hypothesis Tracking
MIMO	multiple-input multiple-output
mmWave	millimeter-wave
MOTA	Multiple Object Tracking Performance Accuracy
MOTP	Multiple Object Tracking Performance Precision
MTI	moving target indicator
NN	neural network
NN-JPDA	nearest-neighbors joint probabilistic data-association
NN-JPDAF	Nearest-Neighbors Joint Probabilistic Data Filter
NTP	network time protocol
RADAR	radio detection and ranging
RCS	RADAR cross section
Re-Id	re-identification
RF	radio frequency
ROI	region of interest
RX	receiver
SC	sensor-to-center association
SNR	signal-to-noise ratio
SS	sensor-to-sensor association
TC	thermal camera
TCP	Transmission Control Protocol
TO	timing offset
TRN	beam-training

TX	transmitter
UDP	User Datagram Protocol
WELM	weighted extreme learning machine

1 Introduction

Contactless sensing pervades our daily life. One of the most obvious and notable examples are camera-based sensors, which capture visible light and convert it into images. In fact, visual systems are key components of a wide range of technologies, as they allow to obtain, at distance, information regarding the surroundings. However, such systems have several limitations, as they are sensitive to environmental lighting and to the presence of dust, smoke, and to weather conditions.

With the first experiments in the early XX century, radio detection and ranging (RADAR) technology can overcome these limitations. RADAR systems work by transmitting an electromagnetic (EM) wave through the atmosphere towards a region of interest and by measuring the signal reflected by objects in such region. Modern RADARs adopt radio waves with frequencies between 300 MHz and 300 GHz, with a wavelength much longer than that of visible light, thus penetrating elements that limit visual systems. Moreover, because of the different principle of operation and frequency bands, RADARs operate independently of illumination conditions.

While, in the past, RADAR sensing was mostly used for military applications such as aircraft or ship surveillance, with the development of technology its range of application became much broader, including complex human sensing scenarios. Technology advancements were two-fold. On the one hand, more efficient radio frequency (RF) front ends were designed, allowing transceivers to operate in the millimeter-wave (mmWave) frequency band (30–300 GHz). This spectrum region is less crowded than lower ones, granting the use of large bandwidth (BW) signals that boost sensing capabilities. On the other hand, the form factor of antennas working at mmWaves is much smaller, as it is related to the wavelength of the signal to transmit, and enabled the design multiple-input multiple-output (MIMO) RF front ends featuring large antenna arrays. The combination of higher frequencies, broader bandwidth, and arrays featuring numerous antenna elements makes mmWave RADAR devices particularly apt for human sensing at short distance. High frequencies result in an increased sensitivity to Doppler shifts, range resolution improves with the increase of signals bandwidth, and MIMO transceivers allow to identify the angle of arrival of the reflections, thanks to the phase differences of the received radio waves. In the last years, due to the employment in wireless communications of hardware featuring the aforementioned characteristics, the cost of such electronic components decreased, easing mmWave RADARs commercial availability and favoring the development of applications involving their use. Many works have explored the possibilities offered by the use of *single* mmWave RADAR sensors, including estimation of human vital signs [1], [2], people tracking and identification [3], [4], activity and gesture recognition [5], [6], but very few considered their use in a cooperative fashion [7], [8]. However, we argue that only the combination of *multiple* sensors can provide the sensing capabilities to face the complexity of many real-life scenarios.

In this thesis, we will focus on collaborative sensing techniques involving the use of *multiple* mmWave RADARs and featuring their fusion with other sensors, in a *multimodal* setting. The aim is to enhance and complement the sensing capabilities by combining more sensors together.

1.1 Sensing with mmWave RADAR networks

While providing appealing range and angular resolution for human sensing applications, commercial mmWave RADARs have limited range (up to 8–10) and are subject to occlusion [3]. This may constitute a significant drawback in crowded spaces or in the presence of walls and furniture. To mitigate for this, it is possible to deploy *multiple* RADAR sensors and illuminate the monitored area from different points of view. By connecting the RADAR devices, they can share information and enhance the sensing performance of the overall system, which constitutes a *RADAR network*. In most cases, the data fusion process requires the position and orientation of the RADAR devices to be known, in order to refer the received information to the same reference system. The most straightforward approach is to manually measure such quantities. However, typical human sensing scenarios often require RADAR sensor positions to be changed and adjusted across time, rendering manual measurements less appealing. Moreover, when envisioning RADAR network deployments in large buildings or areas, involving hundreds or thousands of devices, a manual configuration becomes even less practicable. Therefore, the design of methods to automatically estimate the relative position and orientation of the RADAR devices is key to make feasible the practical deployment of RADAR networks in real-world scenarios.

The first part of this thesis is devoted to building RADAR networks and developing algorithms for their practical implementation. A method for the automatic *self-calibration* of RADAR networks is presented in Chapter 2. It estimates the relative position and orientation of RADAR sensors exploiting the trajectories of people moving freely in the environment, with no need for predefined trajectories and no human intervention. Then, Chapter 3 describes an experimental testbed for the easy deployment and testing of mmWave RADAR networks, comprising a software package for the handling of the network and showing a real implementation with commercial off-the-shelf hardware components.

In the second part of the thesis, we develop methods for data fusion in the context of collaborative mmWave systems. In Chapter 4, the work of Chapters 2 and 3 is exploited to build a RADAR network-based fused tracking system. Each RADAR of the RADAR network is endowed with resource-constrained computational capabilities and performs people tracking independently. Then, each component of the RADAR network shares its tracking information with a *fusion center* that fuses edge nodes' data providing an enhanced, unified tracking among all sensors. The system is designed to work in *real time* and with a low network overhead, as only lightweight track information is shared. Then, in Chapter 5, we investigate the fusion of data from a mmWave RADAR and from an infrared thermal camera (TC). The aim is to jointly perform people tracking and body temperature estimation at distance. The work is motivated by the necessity of timely detect elevated skin temperature to prevent the spread of contagious diseases. Moreover, people tracking enables the possibility of contact tracing, to keep track of the subjects that may have contracted the disease when meeting the infected person. The system includes a deep learning-based re-identification method to identify people based on their gait. The re-identification algorithm works *on-the-fly* without requiring to have in advance any data of the tracked people. It leverages an offline-trained gait features extractor with the addition of a final stage that adapts to new people, once they enter the monitored area. Moreover, a data association technique is developed to associate the temperature estimates from the TC with the tracks from the RADAR and RADAR estimates are used to correct the thermal readings, which are affected by variations in the measurement distance. Finally, Chapter 6 moves a first step towards RADAR networks in an integrated sensing and communication (ISAC) setting. In particular, Wi-Fi packets are exploited to perform RADAR-like sensing. It presents a prototype for multistatic asynchronous ISAC using IEEE 802.11ay protocol. In this scenario, transmitter (TX) and receivers (RXs) are placed at different locations and have independent hardware. In particular, their local oscillators (LOs) are not synchronized, causing carrier frequency offset (CFO) and timing offset (TO). In the work, the offsets are compensated for by exploiting the line-of-sight (LoS) path between devices and some preliminary results are presented. The idea is to study how person's movement is perceived differently based on their relative position with respect to the devices, with the aim of exploiting spatial diversity in data fusion.

1.2 Preliminaries

In this section, the main working principles of some aspects thoroughly present across the whole thesis are summarized, namely, frequency-modulated continuous wave (FMCW) RADARs and people tracking from mmWave RADAR point clouds. When not specified differently, the notation proposed in this section is to be adopted. In some cases, a different notation will be required and the new notation will be promptly defined.

1.2.1 mmWave FMCW Radar

A MIMO FMCW RADAR allows the joint estimation of the distance, the radial velocity and the angular position of the targets with respect to the RADAR device [9]. It works by transmitting sequences of *chirp* signals, i.e., signals whose frequency is increased over time linearly sweeping a bandwidth B, and analyzing their copies, which are reflected back from the environment. A full chirp sequence, termed *radar frame*, is repeated with period T seconds.

Distance, velocity and angle estimation

Typically using discrete Fourier transform (DFT) techniques, the distance, r, of the targets is computed from the measured delay of the reflections, while their velocity, v, is estimated from the frequency shift (also known as *Doppler effect*) induced by the targets' movement. The FMCW RADAR distance resolution is related to the bandwidth B by $\Delta r = c/(2B)$, 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 [3], [10]. Furthermore, using a 2D 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) dimensions, by leveraging phase shifts across different antenna elements. The AoA resolution depends on the number of antennas N in the array and on the spacing between the antenna elements. For the azimuth AoA, it is given by $\Delta\theta = \lambda/(Nd\cos\theta)$. This enables the localization of the targets in the physical space.

Radar detection

The raw output of the RADAR is typically high dimensional for mmWave devices, due to the high resolution. To sparsify the signal and perform a detection of the main reflecting points, a typical approach is the *constant false alarm rate* (CFAR) algorithm [11], which consists of applying a dynamic threshold on the power spectrum of the output signal. Since we are interested in tracking moving targets, a further processing step is required to remove the reflections from static objects, i.e., the *clutter.* This operation is performed using a *moving target indicator* (MTI) high pass filter that removes the reflections with Doppler frequency values close to zero [11].

Radar point-clouds

After the detection phase, a human presence in the environment typically generates a large number of detected points. This set of points, usually termed RADAR *point cloud*, can be transformed into the 3-dimensional Cartesian space (x, y, z) using the distance, azimuth and elevation angles information of the multiple body parts. In addition, the velocity of each point is also retrieved, along with the strength of the corresponding signal reflection.

In this thesis, we always considered as output of mmWave RADARs the corresponding point clouds, as they are directly provided by the adopted commercial devices. Therefore, most of the presented algorithms and solutions are based on this kind of data. The point cloud outputted by the RADAR at frame k is referred to as $\mathcal{P}_k$ and contains a variable number of reflecting points. Each point, $\boldsymbol{p} \in \mathcal{P}_k$, is described by vector $\boldsymbol{p} = [x, y, z]^T$, that identify its 3D coordinates. Moreover, as mentioned, for each point, information regarding its velocity, v_x, v_y, v_z, and the received power P^{RX}, is available, and can be used for specific applications.

1.2.2 People Detection and Tracking

The common approach to people tracking from mmWave RADAR point clouds includes:
(i) **detection**: using density-based clustering to separate the points generated by the subjects from clutter and noise;
(ii) **tracking**: applying Kalman filtering techniques on each cluster centroid to track the movement trajectory of each subject in space.

Regarding *(i)*, detection is typically performed using density-based spatial clustering of applications with noise (DBSCAN) algorithm, an unsupervised density-based clustering algorithm that takes two input parameters, ε and m_{pts}, respectively representing a radius around each point and the minimum number of other points that must be inside such radius to satisfy a certain density condition. Given the description of the RADAR measurements from Section 1.2.1, the coordinates of the points in the horizontal plane (x, y) are used as input for DBSCAN, which outputs a list of detected clusters and a set of points which are classified as *noise*. Typically, the centroid of each cluster is used as an *observation* of the subject's position, feeding a subsequent Kalman filter (KF) tracking algorithm.

In *(ii)*, the KF *state* of each subject at time k is defined as $\boldsymbol{s}_k = [x_k, y_k, \dot{x}_k, \dot{y}_k]^T$, containing the (x, y) subject's coordinates and the corresponding velocities. The state evolution is assumed to obey $\boldsymbol{s}_k = \boldsymbol{A}\boldsymbol{s}_{k-1}$, where the transition matrix $\boldsymbol{A}$ represents a constant-velocity (CV) model. The KF computes an estimate of the state for a target subject at time k, denoted by $\hat{\boldsymbol{s}}_k$, by sequentially updating the predictions from the CV model with the new observations. The association between the new observations (time k) and the previous states (time $k-1$) exploits the nearest-neighbors joint probabilistic data-association (NN-JPDA) algorithm.

1.3 Outline of the Thesis

The reminder of the thesis is organized in two main parts.

The first part describes how to build RADAR networks. Chapter 2 presents a self-calibration method that allows to automatically estimate the relative position and orientation of multiple RADARs exploiting the trajectory of people moving freely in the environment. Then, Chapter 3 presents an experimental testbed for the deployment and testing of RADAR networks. This part incorporates contributions from [12] and [13].

The second part deals with the design of algorithms for data fusion in the context of mmWave monitoring systems. Chapter 4 considers the probem of fusing multiple RADARs to provide a unique and unified tracking among all sensors. In Chapter 5, the cooperation of mmWave RADARs with a thermal camera is addressed, while Chapter 6 moves a first step towards the use of RADAR networks in the context of ISAC. The second part is based on [14], [15], and [16].

Part I

Building RADAR Networks

2

Self-Calibration of mmWave Radar Networks from Human Movement Trajectories

In this chapter, we start our discussion regarding practical solutions to enable the utilization of RADAR networks in real-world conditions. In particular, we develop a method for the *self-calibration* of a mmWave RADAR network that allows to automatically estimate the relative position and orientation of the RADARs using only the trajectories of people moving freely in their field of view (FoV).

2.1 Introduction and related work

The increasingly growing interest in the use of mmWaves for human tracking [3], [4] and activity recognition [17], [18] demands solutions to improve the usability and practicality of such systems. FMCW RADARs working in the mmWave band have emerged as valid alternatives to cameras for indoor monitoring, as they are robust to changing and poor lighting conditions and do not raise privacy concerns [19]. However, commercial mmWave RADARs have limited range [4] (up to 6-8 m) and are subject to occlusion [3]. Covering medium-to-large indoor spaces thus requires multiple RADARs (radar networks), with *known* position and orientation. This raises the problem of how to automatically obtain the positions and orientations of the RADARs, as it is often impossible to know them in advance, and it is impractical to manually input these settings at deployment time. This problem remains unsolved in the existing literature. To the best of our knowledge, only two works have tackled it, i.e., [20], [21]. In [20], the walking trajectory of a single person moving along a straight line (estimated via a KF [22]) is used to compute the relative position and orientation of two RADARs. Even though results are accurate (< 10 cm position and $< 1°$ orientation errors), the system has limitations, as *(i)* only straight-line human movement trajectories are supported, which is unrealistic, and *(ii)* it only works when a single person is being tracked. Both aspects require that the RADAR network performs a dedicated calibration phase in controlled conditions, which is impractical and time-consuming. The authors of [21] propose a similar algorithm for multiple sitting subjects. Here, data association between different RADARs is accomplished by pairing the reflections from the same target seen by two RADARs. The pairing employs a measure of similarity between the respiration waveform of a subject. While this method tackles the multiple target problem, it only works when the subjects are stationary, significantly limiting its application scope.

In this work, we propose mmSCALE, a practical algorithm that automatically estimates the locations and orientations of multiple RADARs with respect to a reference coordinate system by leveraging the movement trajectories of people in the environment. Our method solves the problem of handling free movement trajectories and multiple subjects using *(i)* a standard KF-based tracking routine, *(ii)* an efficient, singular value decomposition (SVD) least-squares (LS) approach to obtain the relative position and orientation of an arbitrary number of mmWave RADARs, and *(iii)* the Hungarian algorithm, applied to an originally designed cost function, to associate the tracks detected by different RADARs. The original contributions of the present work are

1. We propose mmSCALE, a fully automated method for the self-calibration of multiple mmWave RADARs. The algorithm estimates the relative positions and orientations of the RADARs with a median error of 0.18 m and 2.86°, respectively, using the information obtained by tracking people moving freely in the FoV of multiple RADARs.

2. mmSCALE is highly practical and requires no controlled conditions for the calibration. It can support an arbitrary number of RADARs. Thanks to a novel data association cost function, it handles multiple moving subjects in the environment even when occlusions occur. Its fast convergence time enables accurate calibration in less than 6 seconds.

3. We evaluate our method via an extensive measurement campaign involving up to 4 commercial mmWave RADARs and multiple subjects, deployed in realistic conditions including possibly challenging human motion.

The remainder of this chapter is organized as follows: Section 2.2 formalizes the self-calibration problem; Section 2.3 presents and discusses our approach; Section 2.4 describes the experimental results on our testbed; finally, we draw concluding remarks in Section 2.5.

2.2 Problem outline

In this work, we tackle the problem of automatically estimating the position and orientation of multiple RADAR devices on the *azimuth* (horizontal) plane with respect to a common reference system by leveraging the movement trajectories of one or more targets moving in the RADARs' FoV.

2.2.1 System self-calibration

Assume S RADARs are deployed in the space and call F_i, $i = 1, \dots, S$, their reference systems (RSs). Each RS is composed of a pair $F_i = (\mathbf{t}_i, \mathbf{R}_i)$, where $\mathbf{t}_i = [x, y]^T$ represents the position of the origin of the i-th RS and $\mathbf{R}_i$ corresponds to the 2×2 rotation matrix identifying its orientation.

We elect the RS of one RADAR to be the reference RS, and refer all other RSs to the reference one. Therefore, the common RS is known, whereas all the others are unknown. In particular, assuming F_1 is chosen as the reference device, it holds that $F_1 = (\mathbf{t}_1, \mathbf{R}_1)$, with $\mathbf{t}_1 = [0, 0]^T$ and $\mathbf{R}_1 = \mathbf{I}_2$, the 2×2 identity matrix. Hence, our objective is to estimate F_i, $i = 2, \dots, S$, since F_1 is known.

Consider, now, a set $\mathcal{Q} = \{\mathbf{Q}_1, \dots, \mathbf{Q}_S\}$ of S tracks obtained by synchronously tracking the same target at each RADAR device. Each track is a matrix containing the position vectors of the target across time, as seen from a different perspective (i.e., from a different RADAR). Denote by $\mathbf{q}_k$ the (x, y) position of the target (i.e., the first two components of the KF state estimate $\hat{\boldsymbol{s}}_k$) with respect to F_1 at time k, and with $\mathbf{u}_k$ the (x, y) position of the target with respect to the generic RS F_i, at the same time instant. It holds that

$$\mathbf{q}_k = \mathbf{R}_i \mathbf{u}_k + \mathbf{t}_i, \tag{2.1}$$

where $\mathbf{R}_i$ and $\mathbf{t}_i$ represent, respectively, the rotation matrix and translation vector to move from F_i to F_1. Considering a number P of consecutive pairs of estimated positions in tracks $\mathbf{Q}_1 \in \mathbb{R}^{2\times P}$ and $\mathbf{Q}_i \in \mathbb{R}^{2\times P}$, we get an over-determined system of $2P$ independent equations corrupted by noise, due to the imperfect estimation of the positions by the KF. The desired pair $(\mathbf{R}_i, \mathbf{t}_i)$ can be found solving the LS problem

$$(\mathbf{R}_i, \mathbf{t}_i) = \underset{\substack{\mathbf{R}_i \in SO(2), \\ \mathbf{t}_i \in \mathbb{R}^2}}{\operatorname{argmin}} \sum_{k=1}^{P} ||(\mathbf{R}_i \mathbf{u}_k + \mathbf{t}_i) - \mathbf{q}_k||_2, \ i = 2, \dots, S, \tag{2.2}$$

where $SO(2)$ denotes the special orthogonal group in dimension 2 and $||\cdot||_2$ is the Euclidean norm. In Section 2.3, we present an algorithm to efficiently solve this problem.

2.2.2 Challenges of self-calibration with mmWave real data

The automatic self-calibration of a RADAR network using targets moving in their FoV in a real-world scenario poses several challenges concerning *(i)* the presence of multiple tracks, *(ii)* the time synchronization of the RADARs, *(iii)* the association between the tracks, forming pairs to be used for the calibration, and *(iv)* how to assess the quality of the calibration once deployed in a previously unseen, uncontrolled scenario. All these points are briefly discussed below.

Multiple tracks. While in Section 2.2.1 we only considered the case of every RADAR providing only one track, in a real-world scenario, every device would likely provide a list of tracks. This is due to the presence of multiple targets, spurious ghost targets related to mmWave reflections on highly reflective surfaces, or the splitting of the trajectory of a single target into multiple

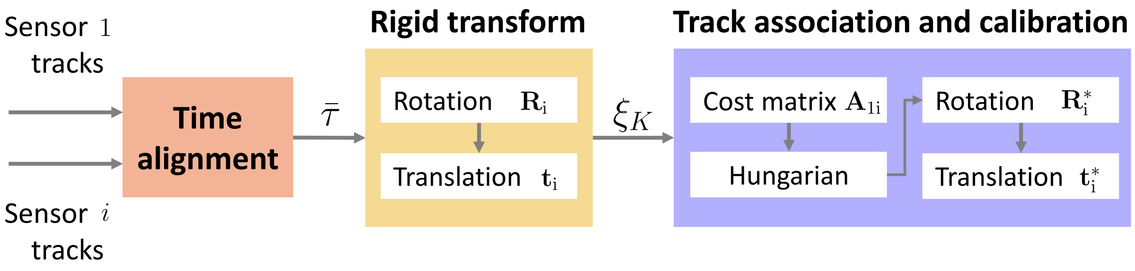

Sensor 1 tracks
Sensor i tracks
Time alignment
$\bar{\tau}$
Rigid transform
Rotation $\mathbf{R}_i$
Translation $\mathbf{t}_i$
ξ_K
Track association and calibration
Cost matrix $\mathbf{A}_{1i}$
Hungarian
Rotation $\mathbf{R}_i^*$
Translation $\mathbf{t}_i^*$

Figure 2.1: mmSCALE worflow.

tracks caused by occlusion events. Thus, we need a method to understand the correspondence between the tracks acquired by different RADARs. Moreover, not all the track pairs provide the same calibration quality. For instance, longer and accurately synchronized tracks are expected to perform better. In the following, focusing on a single time instant, we denote by N_1 the number of tracks from RADAR 1 and by N_i the number of tracks from RADAR i, indicized by $\ell = 1, \dots, N_1$ and $m = 1, \dots, N_i$, respectively.

Time synchronization. Some level of time synchronization between RADARs is needed, as it allows us to associate the pairs $(\mathbf{q}_k, \mathbf{u}_k)$ to be used in Eq. (2.2), which are obtained by different devices. To this end, once a data frame is acquired by one RADAR, we attach a timestamp and then use the timestamps to match data coming from different devices. We call τ_k^ℓ and τ_k^m the timestamps (in seconds) of the k-th position vector in tracks ℓ (of the reference RADAR 1) and m (of the i-th RADAR), respectively. As we empirically assess in Section 2.4, synchronization on the RADAR frame duration level (typical frame rates range from 10 to 30 Hz) is sufficient for mmSCALE to work, so errors of a few milliseconds can be tolerated.

Assessment of the calibration process. As we envision a system that should generalize to unseen, uncontrolled, real-world scenarios, a procedure to evaluate the calibration quality in such conditions is required. To this end, we leverage the fact that, after calibration, associated tracks belonging to the same target should match closely. The same metric can also be used to verify the validity of the calibration across time and to decide whether re-calibration is required.

As further explained in Section 2.3.3, all these aspects are accounted for in the proposed cost function (Eq. (2.6)).

2.3 mmSCALE workflow

From a high-level perspective, mmSCALE solves the calibration problem and the above challenges by operating in three steps, as shown in Fig. 2.1.

(1) Time alignment. Considering all the possible pairings of the reference RADAR 1 with RADARs $i = 2, \dots, S$, a time alignment between the N_1 tracks maintained by RADAR 1 and the N_i tracks from RADAR i is sought, by minimizing the difference between timestamps τ_k^ℓ and τ_k^m.

(2) Rigid transformation. Using the time alignment from point (1), we solve the problem in Eq. (2.2) for all aligned track pairs, obtaining the corresponding rotation matrices and translation vectors.

(3) Tracks association and RADAR calibration. Radar calibration is performed using the best matching track pairs in terms of time alignment and residual rigid transformation error. The key idea is that track pairs that are well aligned in time and that leads to low residuals in Eq. (2.2) represent the same target, as seen by RADAR 1 and i, respectively. Following this rationale, we design a new track-to-track association cost function and find the one-to-one track association yielding the minimum cost. Finally, the calibration of RADAR i is carried out by solving the rigid transformation problem Eq. (2.2) using all the points in the associated track pairs.

2.3.1 Time alignment

The ℓ-th and m-th tracks from RADAR 1 and i, respectively, are aligned in time by exploiting timestamps that are attached to each frame. This alignment is performed so that every element of track ℓ is associated with the element of track m that minimizes the time difference between the two acquisition instants, reducing the tracks to a common length of K time-aligned positions. Elements of track ℓ that do not have a corresponding element of track m within T seconds (recall that T is the duration of a time frame) are discarded. Due to this last operation, our time alignment procedure selects only the portions of the tracks that are sufficiently well synchronized, in order to avoid performing the rigid transformation on wrongly associated points. We define the *mean time alignment* of the (ℓ, m) pair as

$$\bar{\tau}(\ell, m) = \frac{1}{K} \sum_{k=1}^{K} |\tau_k^{\ell} - \tau_k^{m}|. \tag{2.3}$$

The value of $\bar{\tau}(\ell, m)$ represents the quality of the time alignment between the two tracks, and will be used in the track association step (see Section 2.3.3).

2.3.2 Rigid transformation

Let the subject's trajectories observed from the reference RADAR 1 and the i-th RADAR (after time alignment) respectively be $\mathbf{Q}_1 \in \mathbb{R}^{2\times K}$ and $\mathbf{Q}_i \in \mathbb{R}^{2\times K}$. Then, the transformed trajectory $\mathbf{Q}_{i1}$ is given by $\mathbf{Q}_{i1} = \mathbf{R}_i\mathbf{Q}_i + \mathbf{t}_i$. We use the subscript $_{xy}$ to represent a transformed trajectory from the RS of RADAR x to that of RADAR y. To find the optimal $\mathbf{R}_i$ and $\mathbf{t}_i$, we solve the LS problem in Eq. (2.2), that minimizes the square of the Euclidean norm between $\mathbf{Q}_{i1}$ and $\mathbf{Q}_1$. The solution is obtained, in closed form, as [23]

$$\mathbf{t}_i = \bar{\mathbf{q}}_1 - \mathbf{R}_i\bar{\mathbf{q}}_i \ , \text{ with } \mathbf{R}_i = \mathbf{U}\mathbf{V}^T \ , \tag{2.4}$$

where, $\bar{\mathbf{q}}_1$ and $\bar{\mathbf{q}}_i$ are the mean positions of the trajectories $\mathbf{Q}_1$ and $\mathbf{Q}_i$, respectively, and the columns of $\mathbf{U}$ and $\mathbf{V}$ are the left and right singular vectors obtained after the SVD of the covariance matrix $\mathbf{X}\mathbf{Y}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Here, $\mathbf{X} = \mathbf{Q}_i - \bar{\mathbf{q}}_i$ and $\mathbf{Y} = \mathbf{Q}_1 - \bar{\mathbf{q}}_1$. The translation vector obtained after the rigid transformation gives the position of RADAR i in the RS of the reference RADAR. We can also compute the orientation angle of RADAR i using the $\mathbf{R}_i$ matrix. This orientation angle, defined by θ_i, is given as $\theta_i = \cos^{-1}\left(\frac{1}{2}tr(\mathbf{R}_i)\right)$, where $tr(\cdot)$ is the trace of the matrix. mmSCALE computes the rigid transformation parameters $(\mathbf{R}_i^{(\ell,m)}, \mathbf{t}_i^{(\ell,m)})$ for all the aligned (ℓ, m) track pairs from RADARs 1 and i, $i = 2, \dots, S$, as part of the following association step.

2.3.3 Tracks association and RADAR calibration

Tracks association. Our data association strategy consists in *(i)* computing a *cost* for each association $(\ell \leftrightarrow m)$, and *(ii)* solving the resulting combinatorial cost minimization problem to associate the best matching track pairs. Since we operate in real-world conditions, as mentioned in Section 2.2, our cost function needs to incorporate different aspects: *(a)* the length of the tracks, as longer tracks are assumed to provide a better calibration; *(b)* the time alignment of the tracks, as we should compare positions acquired almost simultaneously by the different RADARs; and *(c)* the quality of the rigid transformation, in terms of residual error in superimposing tracks form the different RADARs. We define

$$\xi(\ell, m) = \sum_{k=1}^{K} || \, \mathbf{q}_k^\ell - \mathbf{R}_i^{(\ell,m)}\mathbf{u}_k^m - \mathbf{t}_i^{(\ell,m)} \, ||_2, \tag{2.5}$$

where $\mathbf{q}_k^\ell$ and $\mathbf{u}_k^m$ are the (x, y) positions at time k in tracks ℓ and m, respectively. $\xi(\ell, m)$ is the residual sum of errors in the trajectories after applying the time alignment and the rigid transformation and measures the quality of the solution to the LS problem. Then, the association cost, A, for the tracks pair (ℓ, m), is defined as

$$A(\ell, m) = -\rho(K, \bar{\tau})\,(1 + \xi(\ell, m))^{-1}, \tag{2.6}$$

where $\rho(K, \bar{\tau})$ is a corrective term that depends on the length of the tracks and on their mean time alignment. Recalling that T is the sampling interval of the system, the corrective term is formalized as

$$\rho(K, \bar{\tau}) = \frac{\log(KT)}{1 + \bar{\tau}(\ell, m)}, \tag{2.7}$$

where log denotes the natural logarithm. Note that this corrective term favors tracks which are longer and better time aligned, and penalizes the others. Costs $A(\ell, m)$, $\ell = 1, \dots, N_1$, $m = 1, \dots, N_i$, are arranged into a $N_1 \times N_i$ cost matrix, $\mathbf{A}_{1i}$, and the optimal association of tracks is obtained by minimizing the overall cost, computed through the Hungarian algorithm [24].

Radar calibration. The Hungarian algorithm yields $\min(N_1, N_i)$ pairs of associated tracks, which, according to the mmSCALE rationale, are possibly the same targets seen by the two RADARs. Due to the presence of spurious tracks, ghost targets and clutter, we select a subset of the associated track pairs that have a cost below a threshold A_{th}, which represents a confidence value under which the pair is truly a track pair generated by a human. The N_t track pairs which are selected in this way are then stacked together and used to set up a rigid transformation problem as in Eq. (2.2) that includes all the information available from multiple subjects. The problem is solved with the same procedure described in Section 2.3.2, obtaining the final rotation matrix and translation vector to be used to calibrate RADAR i, namely $(\mathbf{R}_i^*, \mathbf{t}_i^*)$. 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 sub-trajectory.

2.4 Experimental results

We implemented mmSCALE using Texas Instruments IWR1843BOOST mmWave RADARs[*] connected to two NVIDIA Jetson TX2 edge computing devices[†] communicating via Ethernet. The RADARs operate in the 77-81 GHz band in real-time at a frame rate of $1/T = 15$ Hz with a FoV of $\pm 60°$ and $\pm 15°$ over the azimuth and elevation planes, respectively. In this section, we present the experimental results obtained by testing the system in several different scenarios, with 2 and 4 RADAR devices and up to 2 concurrently moving subjects.

2.4.1 Measurements setup and Dataset

To assess the performance of mmSCALE, we conducted tests in a 7×4 m research laboratory (see Fig. 2.2a) equipped with a motion tracking system featuring 10 cameras. This provides the ground truth (GT) 3D localization of a set of four markers placed on each RADAR[‡] with millimeter-level accuracy. We considered 4 scenarios with 2 RADARs and 1 or 2 moving targets, and 3 scenarios with 4 RADARs and 1 moving target. The locations and orientations of the RADARs in the different setups are shown in Fig. 2.2, where the black circles represent the RADARs, the arrows identify their orientations, and the blue area the region of the laboratory where the subjects were allowed to move. We also asked the subjects to move according to 3 possible different trajectories: *(i)* straight, identifying a movement along a straight line, *(ii)* 8-shape, and *(iii)* free, corresponding to an arbitrary trajectory. In the following, we use the notation xR-yT to identify a measurement sequence involving x RADARs and y targets.

2.4.2 Performance evaluation

I. Qualitative results. The input of mmSCALE is a set of tracks obtained from different RADARs, and the output is the estimation of their position and orientation with respect to a common reference system, which, for convenience, we consider to be coinciding with that of RADAR 1. Fig. 2.3 shows a qualitative example of the calibration process with a target moving freely. Here, after finding the optimal rotation and translation parameters, we applied the rigid transformation to the trajectory seen by RADAR 2 (blue line), so as to superimpose it with the one of RADAR 1 (orange line). The transformed trajectory (yellow line) matches well with the reference one, showing a good calibration result. We represent the reference RADAR with a

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

[†]https://developer.nvidia.com/embedded/jetson-tx2

[‡]Using four markers allows us to estimate the orientation angle.

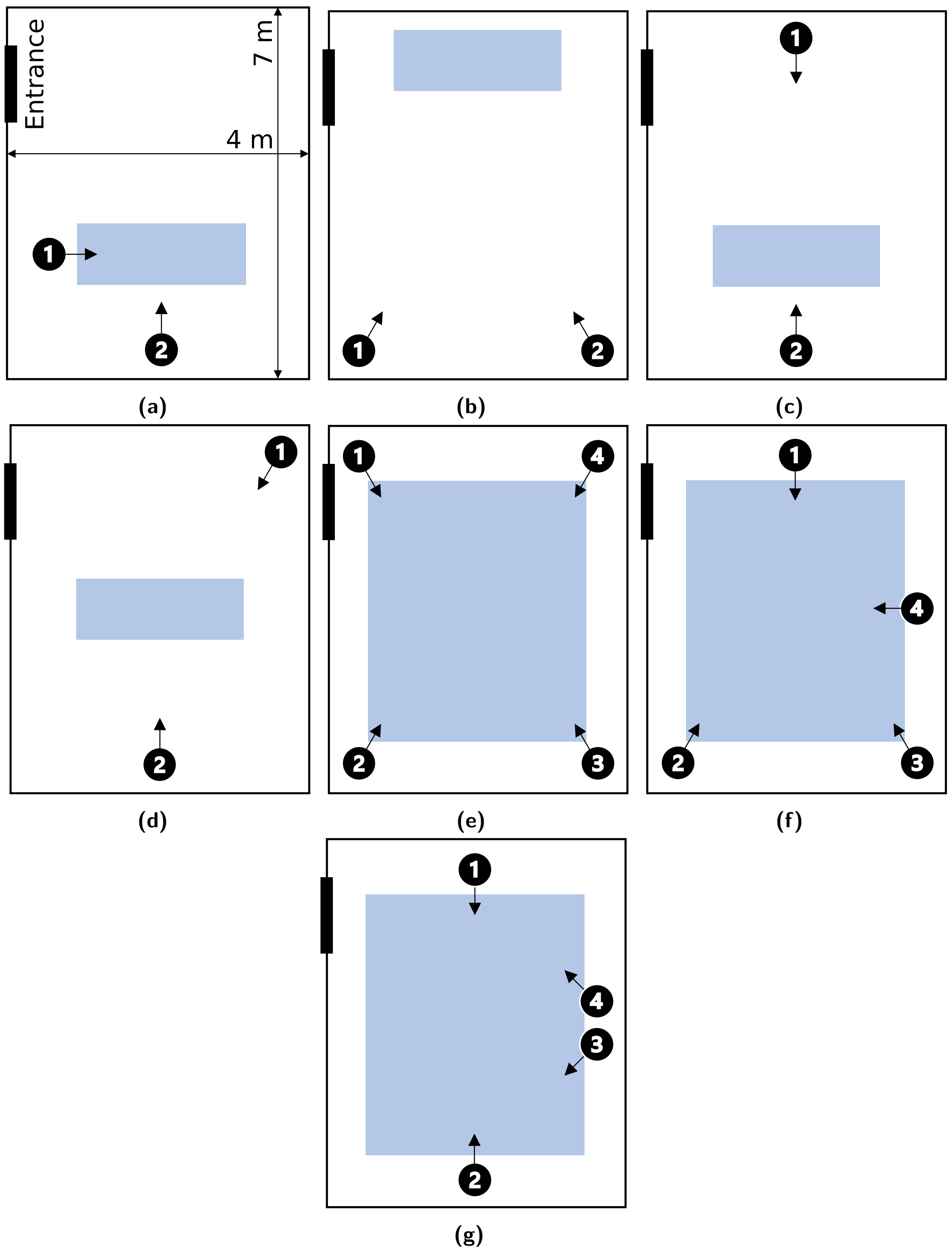


Figure 2.2: Illustration of the setups used to test mmSCALE. The numbered dots represent the RADAR devices, with the arrow identifying the pointing direction of the RADARs. The blue area corresponds to the region where the target moved.

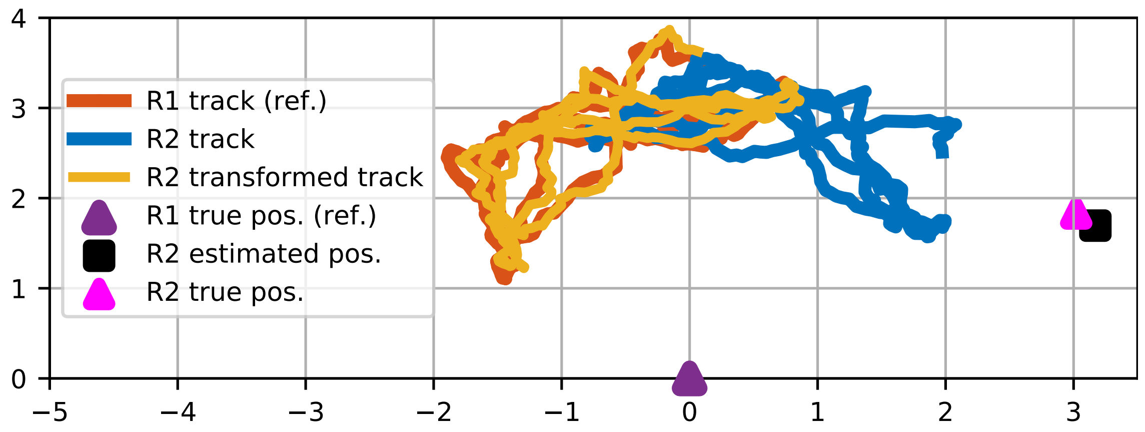


Figure 2.3: Example of the transformation, via mmSCALE, of a free movement trajectory using a 2R-1T setup.

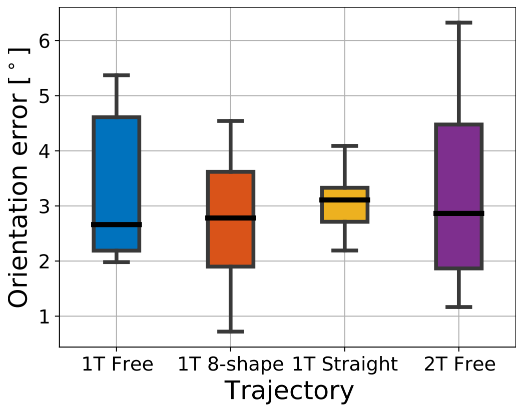
purple triangle (located at $[0,0]^T$), while the black square and the pink triangle mark the estimated position of RADAR 2 and its ground truth, respectively.

II. Position and orientation errors. To evaluate mmSCALE's accuracy in estimating the positions and orientations of multiple RADAR devices, we first consider four 2R-1T and four 2R-2T setups (Fig. 2.2a and Fig. 2.2d), applying our method to calibrate RADAR 2 with respect to RADAR 1. For every setup and every trajectory shape (see Section 2.4.1), we collected two 40 s-long sequences, for a total of 32 sequences. However, since we observed that a time window as short as 6 s is sufficient for our algorithm to converge (see Section 2.4.2-III) we only used the first 6 s of the tracks for all the calibrations.

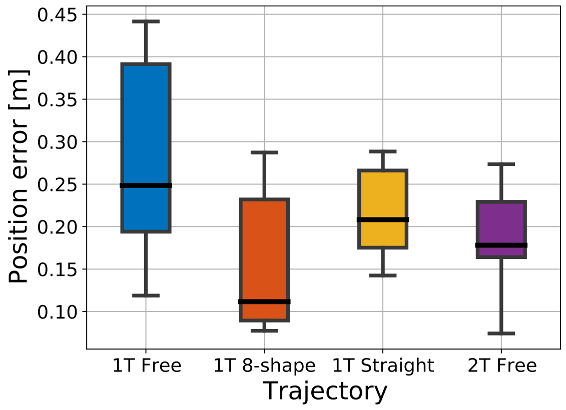
We define the *orientation error* as the absolute value of the difference between the true orientation angle of the RADAR device and the estimated one, which are derived, after calibration, from the rotation matrices, as explained in Section 2.3.2. The *position error* is defined as the Euclidean distance between the estimated position of the RADAR device and its true location.

The calibration performance is summarized in Fig. 2.4, which shows the distribution of the orientation and localization errors for the three trajectory types in a single-target setting (1T), and for the free movement case in a 2 targets setting (2T). Regarding the orientation error, we observe that its median, for all the three trajectories, is about 3°, without a great impact of the shape of the trajectory. On the other hand, the latter influences the RADAR position estimation, where the best results are provided by the 8-shape trajectory, reaching errors as low as 11 cm. The worst case is represented by the free movement, giving a median error of 25 cm.

With 2 concurrently (and freely) moving subjects, mmSCALE obtains similar orientation angle estimation accuracy, with a slightly increased variance, whereas the positioning error significantly improves with respect to the single free moving target case (see also Tab. 2.1). This is a distinctive advantage of the track association process, which can simultaneously exploit all trajectories seen by the two RADARs. As a result, the number of measurements available for the calibration process increases over the same observation period. With 2 subjects, we also empirically assessed suitable values for the threshold parameter A_{thr}. Our results showed that spurious tracks lead to association costs closer to 0 in all the observed cases, whereas complete human trajectories have



(a) Orientation error.



(b) Position error.

Figure 2.4: Orientation and position errors with 2 RADARs and one (1T) or two (2T) subjects on the scene.

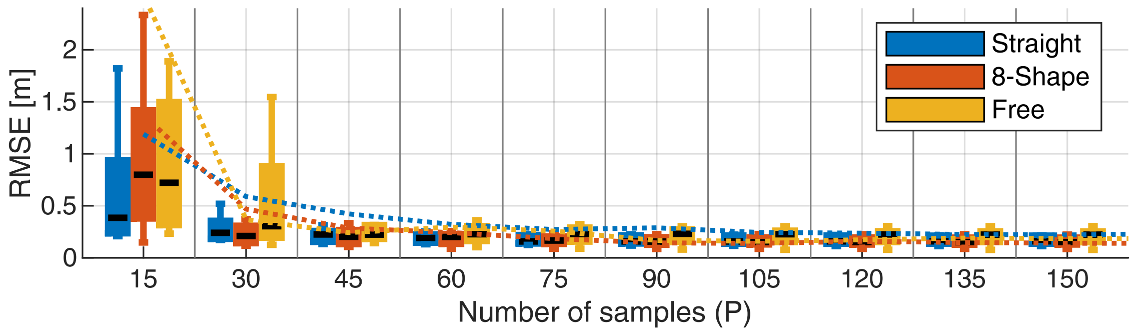
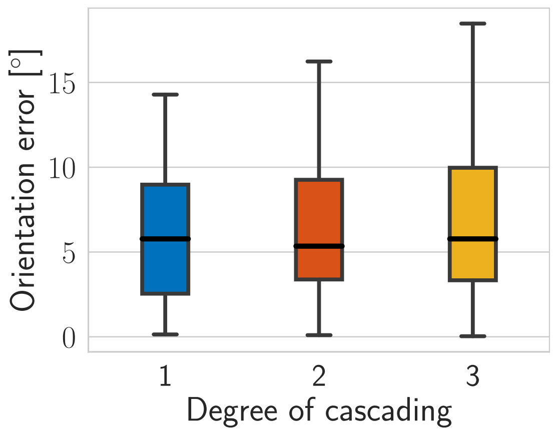


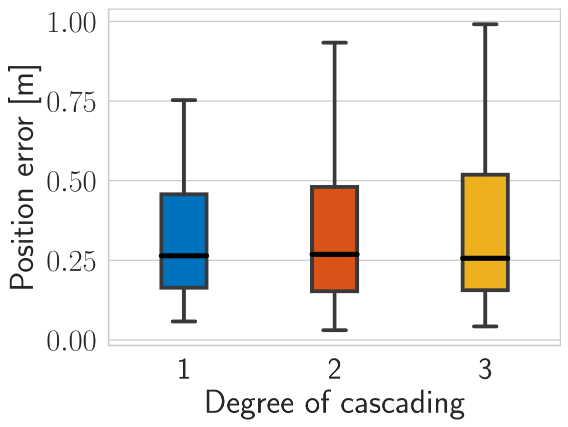
Figure 2.5: Statistical dispersion of trajectory RMSE when calibrating with P points from the trajectory in various 2R-1T setups. The dotted lines represent the median of the RADAR position using P samples.

costs strictly lesser than -2. Hence, we set $A_{\text{thr}} = -2$.

III. Calibration quality assessment. We now examine the number of samples mmSCALE needs in order to achieve a sufficiently low residual calibration error, namely the value of P in Eq. (2.2), and propose a practical method to assess the calibration quality at run time. As a measure of the calibration quality, we use the residual error after transforming the trajectories of RADAR i with respect to RADAR 1 (the reference); if tracks from the two RADARs match well, we can assume the calibration has reached a low positioning and orientation error. This is demonstrated in Fig. 2.5, which shows the distributions of the RMSEs obtained by comparing the transformed track from RADAR i with the one from the reference RADAR 1. To study the impact of the number of samples, we only used the first P trajectory points to compute the rigid transformation and then transformed the entire sequence. P is varied from 15 to 150, which corresponds to a time duration from 1 to 10 seconds. With the dotted curves instead, we show the median error on the estimated position of RADAR i with respect to the GT. In practice, the GT is not available, so only the transformed trajectory RMSE can be computed. We observe that the median RMSE of all the trajectories falls below 0.5 m with just 30 samples, i.e., 2 seconds of data. The RMSE distributions for the straight and 8-Shape trajectory reach a plateau with 45 samples (3 seconds), whereas, for the free trajectory, it takes about 75 samples (5 seconds). The



(a) Orientation error.



(b) Position error.

Figure 2.6: Statistical distributions of all the orientation errors and position errors of the RADARs for different degrees of cascade in the 4R-1T setups.

Table 2.1: Summary of the median errors for different setups

	2 Radar setup				**4 Radar setup**
Median error	**1T Straight**	**1T 8-Shape**	**1T Free**	**2T Free**	**1T Free**
Orientation [°]	3.1	2.78	2.66	2.86	5.6
Position [m]	0.21	0.11	0.24	0.18	0.26

estimation error on the RADAR position also reaches low values within 90 samples (6 seconds), showing an identical trend to the trajectory RMSE. Therefore, 6 seconds are sufficient to assess the quality of the calibration. If needed, a re-calibration can be performed by collecting another set of trajectories and re-applying mmSCALE.

IV. Cascaded calibration. mmSCALE works by calibrating pairs of RADARs, taking one of them as the reference. In practical indoor RADAR networks scenarios, the aim is to have a single global reference system and calibrate multiple RADARs with respect to it. We expect that not all the deployed RADARs have a FoV that partially overlaps with the reference RADAR, i.e., the origin of the global reference system. As a result, direct one-to-one calibration is infeasible. In these cases, mmSCALE can be applied in *cascade*, calibrating each RADAR with respect to another RADAR with which it partially shares the FoV, e.g., in Fig. 2.2e, RADAR 2 with respect to RADAR 1, RADAR 3 with respect to RADAR 2, and so on. We define a *cascade sequence* as the ordered list of the RADAR numbers according to the calibration order ($[1, 2, 3, 4]$ in the previous example from Fig. 2.2e). Furthermore, we refer to the *degree* of a RADAR in the cascade sequence as its position in the list, starting from 0, which is the degree of the reference RADAR. The final calibration with respect to the reference RADAR can be obtained by composing the obtained rigid transformations sequentially. Note that cascaded calibration can occur in any order, as long as adjacent RADARs in the cascade sequence have overlapping FoVs. We applied this technique to the 4R-1T setups shown in Fig. 2.2e-2.2g. We consider all feasible cascading permutations:

e.g., in Fig. 2.2g all permutations involving the calibration of RADAR 4 with respect to 3 (or vice-versa) are infeasible as their 120° FoVs do not overlap. Fig. 2.6 shows the calibration error distributions for RADARs with different degree in the cascade. The median errors are almost unaffected by cascaded calibration, while the variance is slightly higher for higher degrees in the cascade because the errors in θ and $\mathbf{t}$ keep adding up linearly as the degree of the RADAR increases. The "4 Radar setup" section of Tab. 2.1 reports the overall median errors across all the degrees of cascade. The above results show that mmSCALE can keep the cascaded calibration error within reasonable levels in realistic scenarios, where a room would include a network of typically up to 3-4 RADARs.

2.5 Conclusions

In this chapter, we presented mmSCALE, a practical and fully autonomous self-calibration method to estimate the relative position and orientation of multiple mmWave RADARs, by exploiting the tracking of people moving in the environment. We evaluated the system in several realistic conditions, including up to 4 RADARs and 2 subjects, achieving median position and orientation errors of 0.18 m and 2.86°, while still allowing targets to move freely during the calibration process. Moreover, our system requires a calibration window as short as 6 s, handles multiple targets, and is robust to occlusion events. These results suggest that mmSCALE can be a viable candidate to calibrate mmWave RADAR networks for real-time applications.

3
A Testbed for mmWave Radar Networks

In this chapter, we present an experimental testbed that greatly ease the deployment and management of RADAR networks. This allows to speed up experimentation, a fundamental part of research in such field.

3.1 Introduction and related work

While the first works regarding RADAR networks begin to appear [12], [25], in the literature there are still no experimental testbeds for the fast deployment and testing of RADAR network algorithms. Also, simulations have shown to be inapt for the development of human sensing algorithms, as reflections generated by humans moving in real life environments are often too complex to be accurately modeled. In [26], the authors propose a ray tracing model for RADAR signals propagation in RADAR networks, but they neither consider human targets nor the specific case of mmWaves reflections. In [27], a simulator to estimate the RADAR cross section (RCS) of people from computer animated human models is presented. The results are compared with measurements obtained from a real 2.4 GHz RADAR, but only providing a qualitative analysis. Also, the authors use a primitive-based approach which works well only at low frequencies, as proven in [28], where a similar simulator is proposed. In [29], a RCS of pedestrian simulator is proposed and evaluated using a 77 GHz RADAR achieving acceptable results in the estimation of range-time, doppler-time, and range-doppler maps. However, all these works are tested with only one person performing simple movements (e.g., walking on a straight line) and single-input single-output RADARs, while at least a linear array of antennas is required for most applications. Furthermore, these simulators fail to consider some of the crucial aspects of RADAR measurements in real life such as the presence of furniture or walls, introducing noise, clutter, and ghost targets;

the presence of multiple people, producing additional reflections and occlusions; typical real life movements, that produce a wide variety of reflections that are usually not generated by simple movements. In conclusion, currently available simulators are not suitable to develop mmWave RADAR sensing algorithms that will work in practice. For this reason, experimental testbeds are fundamentally important.

The main contributions of this work are:

1. the design of RadNet, an experimental testbed for the easy deployment and testing of mmWave RADAR networks;
2. the implementation of RadNet using commercial edge computers and mmWave RADARs;
3. the implementation of a multi-radar people tracking algorithm, to demonstrate RadNet capabilities.

The remainder of this chapter is organized as follows. Section 3.2 presents the RadNet architecture, its design/working principles and features. Section 3.3 formulates the multi-radar people tracking problem and provides the required preliminaries, while Section 3.4 illustrates the obtained results. Finally, concluding remarks are given in Section 3.5.

3.2 RadNet description

3.2.1 Architecture

RadNet adopts a *server-client* architecture [30]. Each RADAR is directly connected to an *edge* computer (a client node) that is capable of performing local processing and each edge computer is connected via a network connection to a *central* node (a server). The central node controls edge nodes, and the edge nodes serve as distributed computational resources to process RADARs data. However, users do not communicate directly with neither the central node nor the edge nodes. The user interface is provided by an additional *driver* node, which is a transparent entity designed to act as an intermediary for passing control messages to the central node. So, all control actions are provided via the driver node. Also, the driver node can be instantiated and terminated from any machine connected to the network, allowing a flexible control of the system, see Fig. 3.1.

3.2.2 System Control

RadNet works by tasks that are selected by the user through the driver node and assigned to the edge nodes by the central node.

Control Messages

The management of tasks is carried out through a message-passing interface. The aim of the system is to distribute the workload among the edge nodes, while having a centralized unit managing each

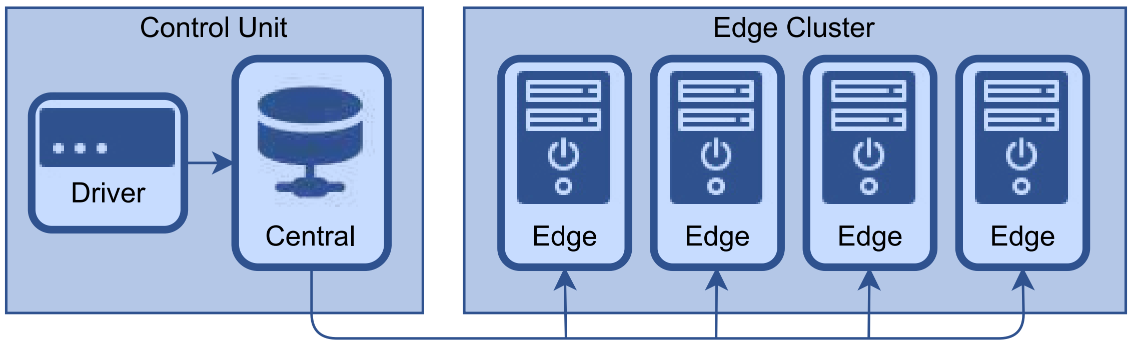


Figure 3.1: RadNet system overview.

of them. The connection between nodes is established and maintained by an internal exchange of messages, which is detailed later in this section.

Currently, available driver commands are:

- *ping*: to measure the latency between central and edge nodes;
- *speed*: to measure the communication speed between central node and an edge node;
- *timesync*: to verify the time synchronization between central node and an edge node;
- *open/close*: to enable/disable edge connections to the system;
- *start* and *stop*: to start and stop RADAR processing at the edges.

Note that additional parameters can be specified for each command, providing more flexibility. Also, the framework allows extending the functionalities provided by the commands and adding new ones.

As previously mentioned, the system works by exchange of messages between the nodes. The driver node only communicates with the central node and the central node distributes the messages to the edge nodes. An overview of the overall system is presented in Fig. 3.1. The control unit manages the system and consists of central and driver nodes. The edge cluster consists of edge nodes providing distributed processing capabilities.

Both the edge nodes and the central node send and receive messages. A message consists of a header, an action, and data, if available. After the retrieval of a message, the receiver performs a certain task and responds with another message that can contain some results related to the current task or a simple acknowledgement. Communication between nodes is always kept alive by checking messages and responding.

Fig. 3.2 shows a typical exchange of messages from the system's startup, which will be thoroughly described next. At the beginning, the central node is initialized. Then, an edge tries to connect to the central node by sending a *join* message. Upon receiving the *join* message, the central node checks the message header and meta data which contains sender's information such as the IP address and the name of the edge node. Then, the central node proceeds to register the edge and stores its information so that, later on, it will be able to communicate with it. If the registration is completed successfully, a *welcome* response is sent to acknowledge the edge about the success of the join operation. At this point, the edge node becomes idle and communicates this via an *idle* message. During the idle period, the edge node stays in receive mode and waits for the assignment of a new action by the central node. In the example, a *start* message is sent to the driver and forwarded to an edge device. This command tells the edge to start some RADAR processing (e.g., to start performing people tracking). Upon receiving *start* message, the edge starts its local process and sends back a *starting* confirmation message, which is received from the central node. Then, as long as the task is not stopped, the central node keeps sending *continue* messages to the edge node, which replies with *processing* messages. Note that *processing* messages sent by an edge node may also contain some data which comes from its processing operations, such as, for example, tracking information or raw point cloud data from the RADAR, depending on the required task. When a *stop* command is issued by the driver, it is forwarded to the intended edge node by the central node and the latter stops its processing operations and acknowledges the reception of the command by answering, in turn, with a *stop* confirmation message. Afterwards, the edge node goes into an idle state.

Parallelization and Data Management

Since applications based on RADAR networks often have to work in an online fashion, delays and latencies must be avoided or reduced as much as possible. For this purpose, RadNet is developed using multi-threading. All system modules which can work independently are deployed in a separate thread. This allows communication tasks and RADAR processes to continue in parallel, without interfering with one another. Also, data management is implemented in parallel to other activities. At an edge node, after retrieving and processing data from the RADAR, these data are placed in a temporary buffer. During processing, the communication module keeps checking the temporary buffer and, as soon as some data are available, they are sent to the central node. Similarly, at the central node, once data are received, a separate thread to save them is started, so that the system can keep working in parallel. Besides being sent, data can be stored locally at the edges or at the central node, or they can be consumed on-the-fly, without being saved.

3.2.3 Implementation Details

The backbone of RadNet framework is written in Python using socket programming and multi-threading. The communication between the nodes in RadNet is carried out via Transmission Control Protocol and Internet Protocol (TCP/IP) [31] as TCP allows data sequencing, guarantees the delivery of the transmissions and provides error checking. Thus, it is reliable and allows a simpler system implementation. In the future, we are also planning to allow users to choose between TCP and User Datagram Protocol (UDP) to send the data, as the latter may provide a faster data stream in certain situations. A dedicated (user defined) error correction protocol for data and commands may have to be implemented in this case. As already mentioned, RadNet has been designed to be modular, so that new functionalities can be added easily.

3.3 Application example

In order to test RadNet on a real application, we implemented a multi-radar multi-target tracking system on it and we used RadNet to evaluate its performance.

In the system, each RADAR is connected via USB to an *edge computer* (corresponding to a RadNet edge node) that performs a local target tracking, while all edge computers are connected via Ethernet to a *Fusion Center* (FC, that is instantiated on the central node). The FC receives tracking information from the edges at regular time intervals and uses them to produce a central unified fused tracking. For the fusion, the RADARs relative position and orientation is required to refer all data to the same reference system. Here, we assume to have them and, in practice, we retrieve them from ground-truth (GT) data. Methods for their automatic estimation can be found in [12], [32].

3.3.1 People tracking notation

The common approach to people tracking from mmWave RADAR point-clouds [3], [4], [33] includes *(i)* a detection phase via density-based clustering algorithms (e.g., DBSCAN [34]) to separate the reflections from multiple subjects, and *(ii)* applying Kalman filtering (KF) techniques [22] on each cluster centroid to track the movement trajectory of each subject in space. In this work, we estimate the subjects' trajectories in the (x, y) horizontal plane. The *state* of each subject at time k is defined as $\mathbf{x}_k = [x_k, y_k, \dot{x}_k, \dot{y}_k]^T$, where $\dot{x}_k$ and $\dot{y}_k$ are the velocity components along the two axes. Also, we assume that the state evolves as $\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1}$, where the transition matrix $\mathbf{F}$ represents a constant-velocity model [35]. Considering RADAR sensor s and target n, we denote by $\hat{\mathbf{x}}_k^s(n)$ and $\mathbf{C}_k^s(n)$, respectively, the KF state estimate and the corresponding 4×4 error *covariance matrix* computed at time k. A track $\mathcal{T}_k^s(n) = \{\hat{\mathbf{x}}_k^s(n), \mathbf{C}_k^s(n)\}$ from sensor s is a pair composed of a state estimate and its covariance obtained at time k. $\mathbf{P}_k^s(n) = (\mathbf{C}_k^s(n))^{-1}$ denotes the *precision matrix* corresponding to the inverse of the covariance matrix $\mathbf{C}_k^s(n)$. In the following, we will use subscript "$k|m$" to denote the same quantities computed at time k using observations up to time $m \leq k$.

3.3.2 Data Fusion

The FC operates in a time-slotted fashion: it considers subsequent time windows and, inside each window, it only processes the information produced during that particular time interval. The behavior of the track fusion algorithm differs in case it has to combine two sensor tracks (SS fusion) or one sensor track and a so-called central track (SC fusion), as we now explain. Considering a specific subject n, if the FC is currently not maintaining a track for it, but one or more RADARs are, a new central track has to be initialized based on the received information from the local sensors. Two cases may happen: *(i)* if n is tracked by one sensor only, then, the corresponding central track is initialized using its state and covariance; *(ii)* if n is tracked by more than one sensor, let us say, sensor 1 and sensor 2, then, their tracks are fused into a single central track (SS fusion). In case *(ii)*, the fusion equations are

$$\mathbf{C}^c_{0|0}(n) = \left[\mathbf{P}^1_0(n) + \mathbf{P}^2_0(n)\right]^{-1}, \tag{3.1}$$

$$\hat{\mathbf{x}}^c_{0|0}(n) = \mathbf{C}^c_{0|0}(n)\left[\mathbf{P}^1_0(n)\hat{\mathbf{x}}^1_0(n) + \mathbf{P}^2_0(n)\hat{\mathbf{x}}^2_0(n)\right], \tag{3.2}$$

where, here and in the rest of the paragraph, superscript c denotes quantities that refer to the FC. This fusion rule is typically used when the errors at the sensors are uncorrelated or the correlation can be neglected [36]. Note that, in case more than two RADARs are tracking the same target, the above process is sequentially repeated, firstly using sensors 1 and 2 to produce a central track, and then fusing the resulting track with the information from sensor 3 and so on until the information from all sensors (RADARs) is used.

If, instead, 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 $\boldsymbol{\mathcal{T}}^c_m$ and $\boldsymbol{\mathcal{T}}^s_m$ the set of tracks maintained by the FC and by sensor s, respectively, at the central time step m. Upon receiving the local information $\boldsymbol{\mathcal{T}}^s_m$, the FC runs a track-to-track association algorithm to find pairs of corresponding tracks $\{\mathcal{T}^c_m(n), \mathcal{T}^s_m(n)\}$. Once such pairs are available, each central track is updated with its corresponding sensor track using the *information decorrelation* method [36], [37] as follows

$$\mathbf{C}^c_{m|m}(n) = \left[\mathbf{P}^c_{m|m-1}(n) + \mathbf{P}^s_m(n) - \bar{\mathbf{P}}^s(n)\right]^{-1}, \tag{3.3}$$

$$\begin{aligned}\hat{\mathbf{x}}^c_{m|m}(n) = \mathbf{C}^c_{m|m}(n)\Big[&\mathbf{P}^c_{m|m-1}(n)\hat{\mathbf{x}}^c_{m|m-1}(n) + \\ &+\mathbf{P}^s_m(n)\hat{\mathbf{x}}^s_m(n) - \bar{\mathbf{P}}^s(n)\bar{\mathbf{x}}^s(n)\Big],\end{aligned} \tag{3.4}$$

where $\bar{\mathbf{P}}^s(n)$ and $\bar{\mathbf{x}}^s(n)$ are the last communicated precision matrix and state estimate of track n from sensor s to the FC. Information decorrelation copes with the problem of correlated tracks between the FC and the RADAR sensors by removing the most recently received information about target n, as, otherwise, this would be accounted for twice by just summing $\mathbf{P}^c_{m|m-1}(u)$ and $\mathbf{P}^s_m(n)$. For more details about the fusion algorithm, including those regarding the time-slotted information processing and the tracks association, please refer to [25].

3.4 Experimental results

We tested RadNet using 4× IWR1843BOOST Texas Instruments mmWave RADARs* operating in the 77-81 GHz band in real-time at a frame rate of $1/T = 15$ Hz with a FoV of ±60° and ±15° over the azimuth and elevation planes, respectively. Each RADAR was connected via USB to an edge computer. As edge computers, we used 1× Jetson TX2 DevKit†, 1× Jetson Nano DevKit‡,

*https://www.ti.com/tool/IWR1843BOOST

†https://developer.nvidia.com/embedded/buy/jetson-tx2-developer-kit

‡https://developer.nvidia.com/embedded/buy/jetson-nano-devkit

Table 3.1: Summary of tracking algorithm performance

Median error	**1 RADAR**	**2 RADARs**	**3 RADARs**	**4 RADARs**
1T-MOTA	96%	97%	96%	94%
1T-MOTP [m]	0.08	0.12	0.15	0.15
2T-MOTA	74%	77%	85%	86%
2T-MOTP [m]	0.10	0.29	0.26	0.22

and 2× Jetson Xavier NX DevKit[§].

3.4.1 Measurement setup and Dataset

To assess the performance of the people tracking algorithm implemented using RadNet, we conducted tests in a 7×4 m research laboratory (see Fig. 3.3a) equipped with a motion tracking system featuring 10 cameras. This provides the GT 3D localization of a set of markers placed on the RADARs and on the subjects with millimeter-level accuracy. We acquired 5 sequences with 1 target (1T) and 30 sequences with 2 targets (2T). Each sequence is 40 seconds long. People either moved according to predefined trajectories or moved freely, as depicted in Fig. 3.3. For single target sequences, only free movement was considered. Subjects were tracked by 4 RADARs simultaneously, deployed in 2 different setups. Then, tracking information have been fused in different ways to evaluate distinct scenarios. In particular, all possible combinations of 1 to 4 RADARs have been considered for the fusion, resulting in $\binom{4}{1} + \binom{4}{2} + \binom{4}{3} + \binom{4}{4} = 15$ different experiments per sequence. Thus, a total of $35 \times 15 = 525$ experiments have been analyzed.

3.4.2 Evaluation metrics

To assess the tracking performance, we adopt the *Multiple Object Tracking Performance Accuracy* (MOTA) metric, 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 [38].

3.4.3 Tracking performance

Tab. 3.1 summarizes the median tracking performance, across all the configurations, see Fig. 3.3. Columns identify the various fusion cases: when only 1 RADAR is used, i.e., there is no fusion; when i RADARs are used, $i > 1$, it means that their tracks have been fused and the results refer to the fused track. Considering the 1T case, there is almost no difference between using a single RADAR or multiple RADARs. This result was expected, since the tracking of a single person in an empty room is a relatively simple task, as no occlusions can occur to impact the quality of RADAR tracks. A slight worsening of MOTA and MOTP is observed as the number of RADARs increases due to the fact that single RADAR target state estimations are not perfect and during

[§]https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit the fusion process this can result in a small increment in the estimation error (the estimation noise is also fused). This is more evident in the MOTP values that go from 0.08 m to 0.15 m, which is still a small error. One may argue that, given these results, in the 1T case the use of only one RADAR is to be preferred with respect to more RADARs. This is reasonable for the very simple case of one target moving in a quite restricted area. If, instead, the target moves in a larger environment, the use of multiple RADARs is required to be able to track it in the entire space, as single sensors range is limited to a few meters. Considering the 2T case, it is possible to observe a clear increase in the MOTA as more RADARs are used for the tracking, going from 74% with 1 RADAR to 86% with 4 RADARs, providing an improvement of 12%. Situations with 2 targets are way more challenging, as they can occlude each other, possibly severely limiting the tracking capabilities of a single RADAR device. MOTP values remains always within a median value of 0.26 m, which is in line with state-of-the-art results [15].

3.5 Conclusions

In this chapter we presented RadNet, the first testbed to facilitate the deployment and testing of algorithms for mmWave *RADAR networks.* We described its architecture and working principles, and implemented a first version of it using commercial mmWave RADARs and edge computers. As an example, we also deployed a distributed multi-radar multi-target people tracking algorithm, showing the improvement that exploiting multiple RADARs can bring, in terms of tracking accuracy (+12%).

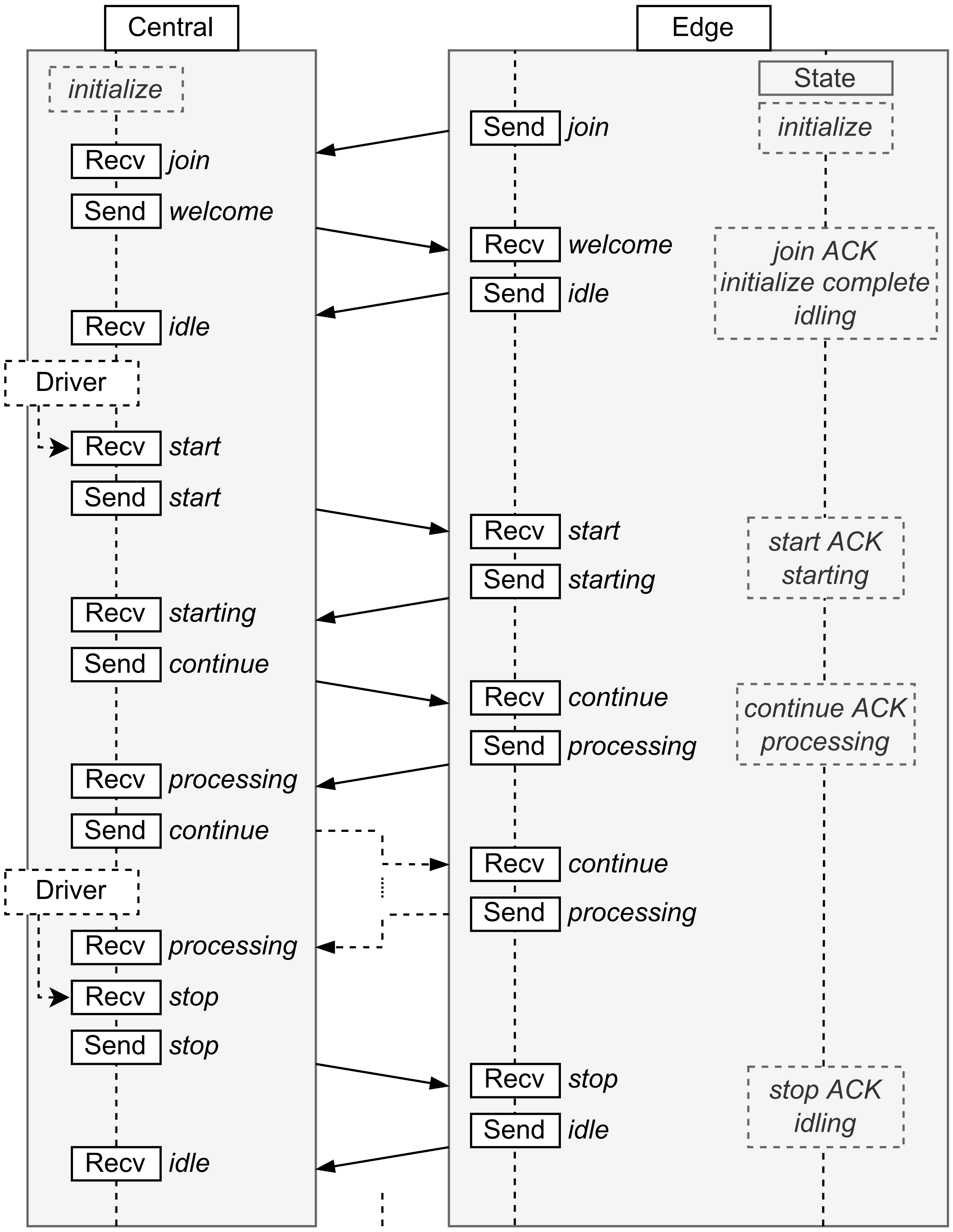


Figure 3.2: Example of communication between central and edge nodes. Send, Recv and ACK annotations in the figure correspond to sending, receiving and acknowledgement events, respectively.

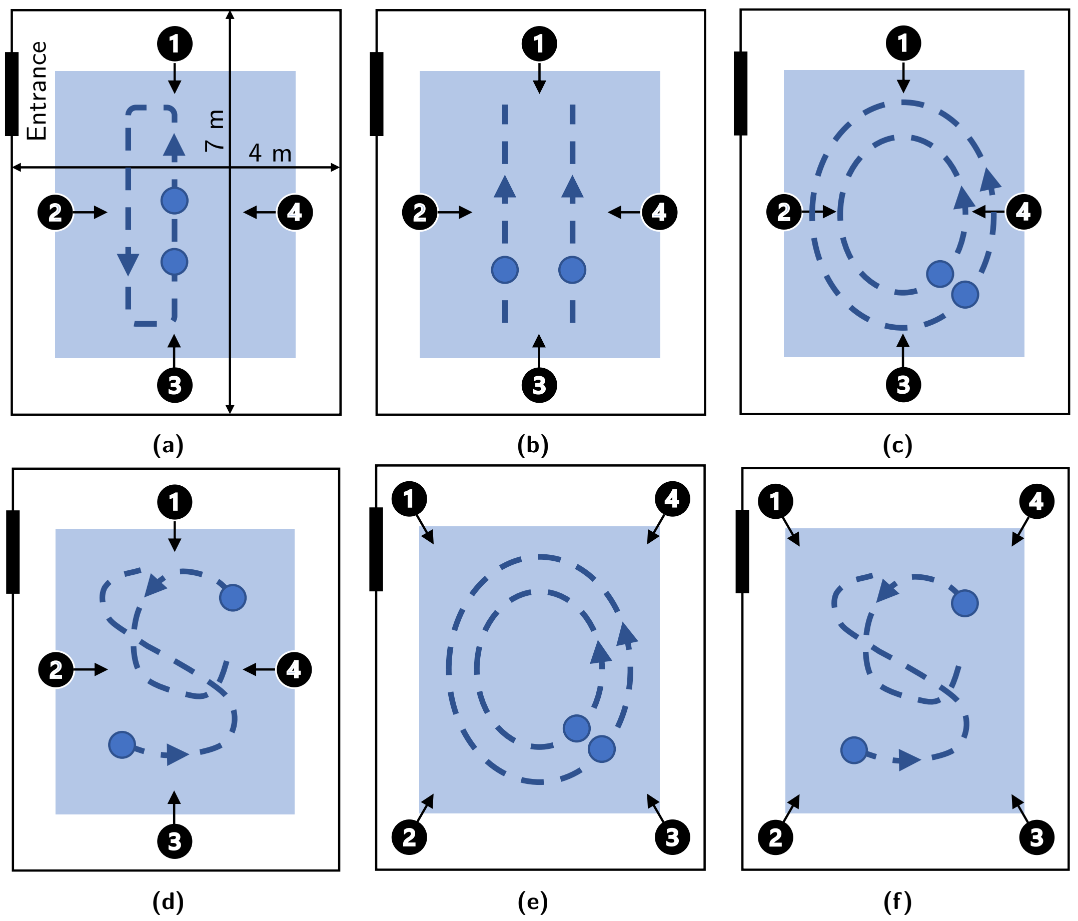


Figure 3.3: Illustration of the setups and trajectories used to test RadNet. The numbered dots represent the RADAR devices, with the arrow identifying their pointing directions. The blue dots represent the subjects, while the dashed blue lines represent subjects' trajectories. Fig. 3.3d and Fig. 3.3f represent free movement.

Part II

Sensing and Data Fusion in Collaborative mmWave Systems

4

Occlusion-Resilient and Self-Calibrating mmwave Radar Network for People Tracking

In this chapter, we start developing algorithms for data fusion with RADAR networks. We leverage the solutions from Chapters 2 and 3 to create a semi-distributed fused tracking systems combining information from multiple mmWave RADARs.

4.1 Introduction

Multistatic RADARs used in, e.g., [39], [40], require synchronizing the devices' clocks in order 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. In this scenario, we need to solve two main issues: *(i)* 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 *(ii)* combine the environment perception capabilities of the multiple RADARs (*sensor fusion*), so as to boost their sensing accuracy and mitigate occlusion. The few existing solutions to point *(i)* present significant practicality and usability limitations in real scenarios [20], [21], [41]. Point *(ii)*, 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 chapter 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 [12]. 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 a fusion center (FC), enhancing the resilience of the tracking process in 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:

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 [12] (presented in Chapter 2) by adding a *masking* phase (see Section 4.4.1) 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 3 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 [13] (see Chapter 3), using 4 commercial mmWave RADARs deployed in realistic conditions and multiple subjects, including challenging human motion. In the most difficult case of 3 subjects moving concurrently, ORACLE achieves a tracking accuracy of 87% and a mean tracking error of 23 cm.

The remainder of this chapter is organized as follows: Section 4.2 provides a summary of the related work. In Section 4.3, the problem tackled by ORACLE is presented and formulated. Section 4.4 presents and discusses ORACLE, the proposed method. Section 4.5 provides some insights regarding the practical implementation of ORACLE and presents the experimental results on our testbed. Finally, concluding remarks are drawn in Section 4.6.

4.2 Related work

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

[42]. Existing works have leveraged this principle for drone detection [43] and people tracking [39], [44]–[46]. 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.

Radar networks A large body of work has considered the use of incoherent RADAR networks outdoors, e.g., in airborne and automotive applications, to improve the detection and tracking capabilities of the standalone sensors, [47]–[50]. 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 occur 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., [47], [48]). 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 [18], [33]. 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 [8], [41]. 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 *(i)* combines the information from *multiple RADARs*, handling the presence of *multiple subjects*, *(ii)* automatically estimates their relative positions and orientations, and *(iii)* shows robust *real-time* performance thanks to its low complexity and distributed computation load.

Radar networks self-calibration Few works have tackled the problem of self-calibration in mmWave RADAR networks, i.e., [8], [20], [21]. These studies share significant practical limitations. [8] proposes a semi-automatic 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. [20], instead, performs calibration using a target of opportunity, but it requires the presence of only one target following a strictly *linear* trajectory in the RADARs FoV. Finally, [21] 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 shape that span a large portion of the FoVs of the RADARs, as they lead to a more accurate calibration.

4.3 Problem outline

In this section, we formalize the problems of combining the information obtained by the different RADARs in the network at a central fusion entity, and of estimating their relative positions and orientations.

4.3.1 Sensor fusion in mmWave RADAR networks

Consider a 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 a FC. The sensors are identified by indices $s = 1, \dots, S$, 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 [35]. At the FC, the state model for target u is

$$\mathbf{x}_m(u) = \mathbf{F}_{T_c}\mathbf{x}_{m-1}(u) + \mathbf{w}_m(u), \tag{4.1}$$

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 $\mathbf{W}$ [3], [36]. The process noise is here considered to be generated by a random acceleration that is not explicitly accounted for by the CV model [3]. Sensor measurements of the state of target n, at time k, are obtained according to

$$\mathbf{z}_k^s(n) = \mathbf{H}\mathbf{x}_k(n) + \mathbf{v}_k^s(n), \quad s = 1, \dots, S, \tag{4.2}$$

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 $\mathbf{V}_k$ [3]. 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 [3].

The aim of our system is to estimate $\mathbf{x}_m$ over time, exploiting the measurements collected by the sensors. Note that: *(i)* 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; *(ii)* the algorithm can handle the fact that the same target may be tracked simultaneously by one or more sensors; and *(iii)* 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 [3], [4], [33] includes: *(i)* a detection phase via density-based clustering algorithms (e.g., DBSCAN [34]) to separate the reflections from multiple subjects; *(ii)* applying Kalman filtering techniques [22] on each cluster centroid to track the movement trajectory of each subject in space. The association between new observations and KF states exploits the Nearest-Neighbors Joint Probabilistic Data Filter (NN-JPDAF) algorithm [51]. 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 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 is able to 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 $\boldsymbol{\mathcal{T}}_k^s = \{\mathcal{T}_k^s(1), \dots, \mathcal{T}_k^s(N_k^s), \tau_k^s\}$ 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. Our aim is 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 $\boldsymbol{\mathcal{T}}_k^s$, $s = 1, \dots, S$. The above problem is complicated by *correlation* between the estimation errors of the tracks obtained at the sensors and at the FC. From Eq. (4.1), 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 [36]. 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, e.g., from 10 to 15 times per second.

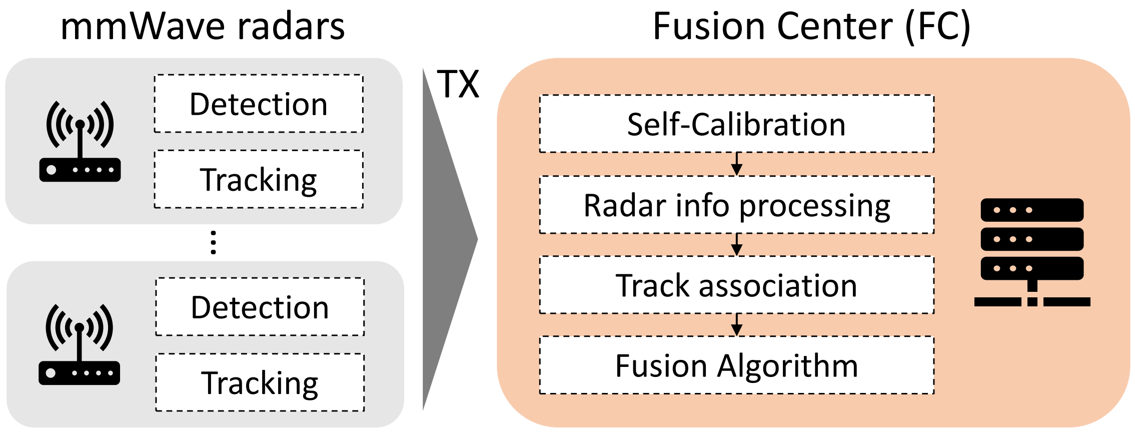


Figure 4.1: Proposed workflow.

4.3.2 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, 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 $\mathcal{F}_s$, $s = 1, \dots, S$, 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 s-th RS's origin and $\mathbf{R}_s$ is the 2×2 rotation matrix specifying its orientation. Without loss of generality, in this chapter we consider a *global* RS (of the FC) which coincides with that of RADAR 1 and for which it holds that $\mathbf{t}_1 = \mathbf{0}_{2\times 1}$ 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 $\mathcal{F}_s$, $s = 2, \dots, S$. 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 $k = 1, \dots, K$, 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 of the translation vector between RADAR s and RADAR 1 (our reference) can be obtained solving the following Least-Squares (LS) problem

$$\underset{\substack{\mathbf{R}_s \in SO(2) \\ \mathbf{t}_s \in \mathbb{R}^2}}{\operatorname{argmin}} \sum_{k=1}^{K} \left|\left| (\mathbf{R}_s \hat{\mathbf{p}}_k^s(n) + \mathbf{t}_s) - \hat{\mathbf{p}}_k^1(n) \right|\right|_2, \tag{4.3}$$

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-dimensional 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 Eq. (4.3), the orientation angle, denoted by θ_s, is given by $\theta_s = \cos^{-1}(\text{trace}(\mathbf{R}_s)/2)$.

In Section 4.4.1 we present the proposed approach to solve the self-calibration problem in the more complex and realistic scenario where: *(i)* multiple sensors concurrently track multiple targets; *(ii)* the track-target correspondence among different sensors is *unknown*, so, an association strategy has to be developed; and *(iii)* the trajectories should be aligned in time before using Eq. (4.3).

4.4 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. 4.1 presents the workflow of ORACLE.
Self-calibration In this phase, the relative positions and orientations of the RADARs are obtained from the trajectories of targets of opportunity (see Section 4.4.1). The steps are:

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 4.4.1).

2. *Track association.* Using the time alignment from point 1), we solve the problem in Eq. (4.3) 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 4.4.1).

3. *Masking.* When estimating the roto-translation parameters at point 4), multiple track pairs will be used all together. To avoid that possibly wrong associations spoil 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 4.4.1).

4. *Radar calibration.* All the track pairs from point 3) are stacked together and used to set up a rigid transformation problem as in Eq. (4.3) that provides the final position and orientation estimates for the RADAR (see Section 4.4.1).

Multi-radar 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 4.4.2). This includes:

1. *Slotted sensor information processing.* The tracks information from the sensors are sent to the FC and processed using a slotted protocol. (see Section 4.4.2).

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 4.4.2).

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 4.4.2).

4.4.1 Self-calibration

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 a 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, in order 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 $\bar{\tau}(\mathbf{n}_1, \mathbf{n}_s) = \frac{1}{K}\sum_{k=1}^{K} |\tau^1_{\mathbf{k}_{1,k}} - \tau^s_{\mathbf{k}_{s,k}}|$, where $\mathbf{k}_{j,k}$ denotes the k-th 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 4.4.1).

Track association

Our data association strategy consists in 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: *(i)* the length of the trajectories, as longer trajectories are assumed to provide a better calibration; *(ii)* the time alignment of the trajectories, as we should compare position estimates acquired almost simultaneously by the different RADARs; and *(iii)* 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

$$A(\mathbf{n}_1, \mathbf{n}_s) = -\rho(K, \bar{\tau})\,[1 + \xi(\mathbf{n}_1, \mathbf{n}_s)]^{-1}\,, \tag{4.4}$$

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 Eq. (4.3) in closed-form through a Singular-Value Decomposition method [52]. 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 a $N_1 \times N_s$ cost matrix and the optimal association of trajectories is obtained by minimizing the overall cost, computed through the Hungarian algorithm [24]. 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 p-th selected track from RADAR q. In our experiments, we empirically adopted $A_{\text{self}} = 18$.

Masking

In phase 4) (see Section 4.4.1), 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 purposedly ignoring (*masking*) some of the track associations in the computation of the self-calibration parameters. The same association cost of Eq. (4.4) 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 Eq. (4.4) 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 element 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 5 track pairs.

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 Eq. (4.3). The problem is solved with the same procedure described in Section 4.4.1 [52], 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 sub-trajectory.

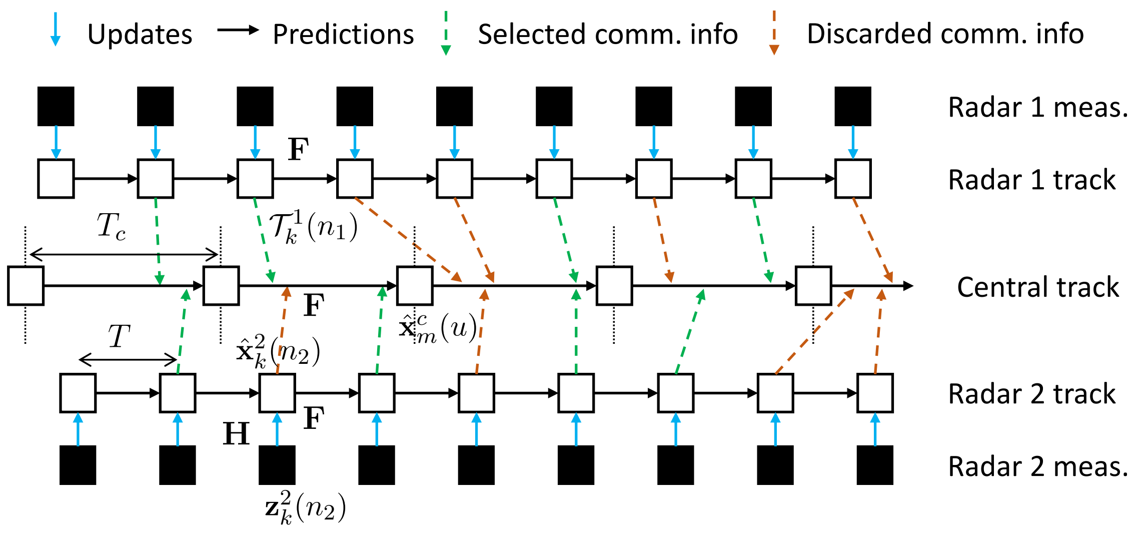


Figure 4.2: Scheme of the proposed fusion algorithm with 2 RADARs.

4.4.2 Multi-radar fusion

Once the RADAR network is calibrated, i.e., 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*, which is defined as the inverse of a covariance matrix, by $\mathbf{P} = \mathbf{C}^{-1}$. The fusion algorithm, represented in Fig. 4.2, is described as follows:

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. In order 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 $\text{argmin}_{\tau_k^s}(|\tau_m^c - \tau_k^s|)$. 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 $\boldsymbol{\mathcal{T}}_m^s$, 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 *(i)* reduces the number of fusion steps the FC carries out, using only the most recent information available from each sensor, and *(ii)* avoids erroneously fusing outdated tracks.

After selecting the sensor tracks, the FC transforms them to match its own reference system, 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 blkdiag$(\cdot)$ returns a block diagonal matrix of its inputs. The transformation and propagation are performed, together, as

$$\hat{\mathbf{x}}_m^s(n) = \mathbf{F}_{\tau_m^c - \tau_k^s} \left[\bar{\mathbf{R}}_s^* \hat{\mathbf{x}}_m^{s'}(n) + \bar{\mathbf{t}}_s^* \right], \tag{4.5}$$

$$\mathbf{C}_m^s(n) = \mathbf{F}_{\tau_m^c - \tau_k^s} \bar{\mathbf{R}}_s^* \mathbf{C}_m^{s'}(n) \left(\bar{\mathbf{R}}_s^*\right)^T \mathbf{F}_{\tau_m^c - \tau_k^s}^T, \tag{4.6}$$

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 Eq. (4.5) and Eq. (4.6), 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

$$\mathbf{C}_{m|m-1}^c(u) = \mathbf{F}_{T_c} \mathbf{C}_{m-1|m-1}^c(u) \mathbf{F}_{T_c}^T + \mathbf{W}, \tag{4.7}$$

$$\hat{\mathbf{x}}_{m|m-1}^c(u) = \mathbf{F}_{T_c} \hat{\mathbf{x}}_{m-1|m-1}^c(u). \tag{4.8}$$

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 *(i) sensor-to-center* associations SC, to verify whether the sensor tracks correspond to any of the maintained central tracks, and *(ii) sensor-to-sensor* associations (SS), only for sensor tracks which did not find a 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, indicized by variables i and j. Note that i and j may refer to two sensor tracks, in case of an SS association, or to a central track and a sensor track, in 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 sensor-to-center track pair that has a correspondence in previous SC associations is examined to verify whether the association still holds. This operation consists in computing the Mahalanobis distance [53] 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

$$\text{d}_M(i,j) = \left(\hat{\mathbf{x}}(i) - \hat{\mathbf{x}}(j)\right)^T \mathbf{P}(i,j) \left(\hat{\mathbf{x}}(i) - \hat{\mathbf{x}}(j)\right), \tag{4.9}$$

$$\mathbf{P}(i,j) = [\mathbf{C}(i) + \mathbf{C}(j)]^{-1} \tag{4.10}$$

where $\mathbf{P}(i,j)$ is the precision matrix inducing the distance. All track pairs for which $\mathrm{d}_M(i,j) \le A_{\mathrm{th}}$ 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. A $M \times N$ cost matrix, $\mathbf{\Lambda}$, 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 Eq. (4.9) 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 $\hat{\mathbf{x}}_{\mathrm{dec}}(q) = \hat{\mathbf{x}}(q) - \bar{\mathbf{x}}(j)$ and $\mathbf{C}_{\mathrm{dec}}(q) = (\mathbf{C}(q)^{-1} - \bar{\mathbf{C}}(j)^{-1})^{-1}$, respectively. $\bar{\mathbf{x}}(j)$ and $\bar{\mathbf{C}}(j)$ are the last communicated state and covariance matrix from track j, respectively. $\hat{\mathbf{x}}_{\mathrm{dec}}(q)$ and $\mathbf{C}_{\mathrm{dec}}(q)$ represent the decorrelated versions of the corresponding quantities, according to the *decorrelation principle* [36], [37]. The decorrelation operation removes the effect of previous fusion events that, otherwise, would affect the computation of the association cost [36].
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 Eq. (4.9), without modifications. Once matrix $\mathbf{\Lambda}$ is available, the minimum total cost association is obtained by using, e.g., the Hungarian algorithm [24], and all associations whose cost doesn't 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 $\mathbf{\Lambda}$ is built, where $\Lambda_{ij} = d_M(i,j)$, as in Eq. (4.9). Since sensor tracks are 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, e.g., the Hungarian algorithm [24]. In our experiments we adopted $A_{\mathrm{th}} = 18$ and $T_{\mathrm{th}} = 1.3 \times T_c$.

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: *(i)* 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; *(ii)* if the FC receives two tracks that can be associated, say $\mathcal{T}^1_m(n_1)$ and $\mathcal{T}^2_m(n_2)$, 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 [36]. 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 *(ii)*, are

$$\mathbf{C}^c_{m|m}(u) = \left[\mathbf{P}^1_m(n_1) + \mathbf{P}^2_m(n_2)\right]^{-1}, \tag{4.11}$$

$$\hat{\mathbf{x}}^c_{m|m}(u) = \mathbf{C}^c_{m|m}(u)\left[\mathbf{P}^1_m(n_1)\hat{\mathbf{x}}^1_m(n_1) + \mathbf{P}^2_m(n_2)\hat{\mathbf{x}}^2_m(n_2)\right], \tag{4.12}$$

for the couple of associated tracks $\mathcal{T}^1_m(n_1)$ and $\mathcal{T}^2_m(n_2)$. Note that, to detect when a SS fusion has to be performed, our algorithm applies the SS association procedure to all the sensor tracks that have *not* been associated to any central track in the current slot. In case more than 2 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 $\boldsymbol{\mathcal{T}}^c_m$ the set of tracks maintained by the FC at the central time step m. Upon receiving the local information $\boldsymbol{\mathcal{T}}^s_m$, the FC runs the track-to-track association algorithm to find pairs of corresponding tracks $\{\mathcal{T}^c_m(u), \mathcal{T}^s_m(n)\}$. Once such pairs are available, if the timestamp associated with $\boldsymbol{\mathcal{T}}^s_m$ is older than T_{th}, then, the same fusion rule of Eq. (4.11) and Eq. (4.12) 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 [36], [37] as follows

$$\mathbf{C}^c_{m|m}(u) = \left[\mathbf{P}^c_{m|m-1}(u) + \mathbf{P}^s_m(n) - \bar{\mathbf{P}}^s(n)\right]^{-1}, \tag{4.13}$$

$$\begin{aligned}\hat{\mathbf{x}}^c_{m|m}(u) = \mathbf{C}^c_{m|m}(u)\Big[&\mathbf{P}^c_{m|m-1}(u)\hat{\mathbf{x}}^c_{m|m-1}(u) + \\ &+\mathbf{P}^s_m(n)\hat{\mathbf{x}}^s_m(n) - \bar{\mathbf{P}}^s(n)\bar{\mathbf{x}}^s(n)\Big],\end{aligned} \tag{4.14}$$

where $\bar{\mathbf{P}}^s(n)$ and $\bar{\mathbf{x}}^s(n)$ are the last communicated precision matrix and state estimate of track $\mathcal{T}^s_m(n)$ from sensor s to the FC. 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.

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 [3]. 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 4.4.2, 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 fusion center. 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 fusion center. 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.

4.5 Experimental results

We evaluated ORACLE through an implementation of the RadNet platform [13] featuring 4 Texas Instruments IWR1843BOOST mmWave RADARs[*] connected to 2 Jetson Xavier NX DevKit[†], 1 Jetson TX2 DevKit[‡] and 1 Jetson Nano DevKit[§], 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 a FoV of $\pm 60°$ and $\pm 15°$ over the azimuth and elevation planes, respectively. Fig. 4.4 shows a picture of the implemented experimental setup.

4.5.1 Implementation notes on numerical stability

In this section, we provide insight into the numerical stability of ORACLE, which are 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 *(i)* non positive definite and *(ii)* ill-conditioned (with very large condition numbers), causing wrong track associations and fusion results because of the spoiled inverse matrices. The solution to point *(i)* 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, non-positive eigenvalue. We obtain the corrected, positive definite matrix $\mathbf{P}_{\text{pos}}$, as

$$\mathbf{P}_{\text{pos}} = \mathbf{P} + (-\lambda_{\min} + \varepsilon)\mathbf{I}, \quad \varepsilon > 0. \tag{4.15}$$

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 *(ii)*, we use Ridge regularization [54] 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 $\mathbf{P}_{\text{pos}}$. Denote by $\text{cond}(\mathbf{P}_{\text{pos}}) = \lambda'_{\max}/\lambda'_{\min}$ the condition number of matrix $\mathbf{P}_{\text{pos}}$, and by δ the regularization parameter. The regularized covariance matrix, $\mathbf{P}_{\text{reg}}$, is obtained as

$$\mathbf{P}_{\text{reg}} = \frac{1}{1+\delta}(\mathbf{P}_{\text{pos}} + \delta\mathbf{I}). \tag{4.16}$$

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

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

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

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

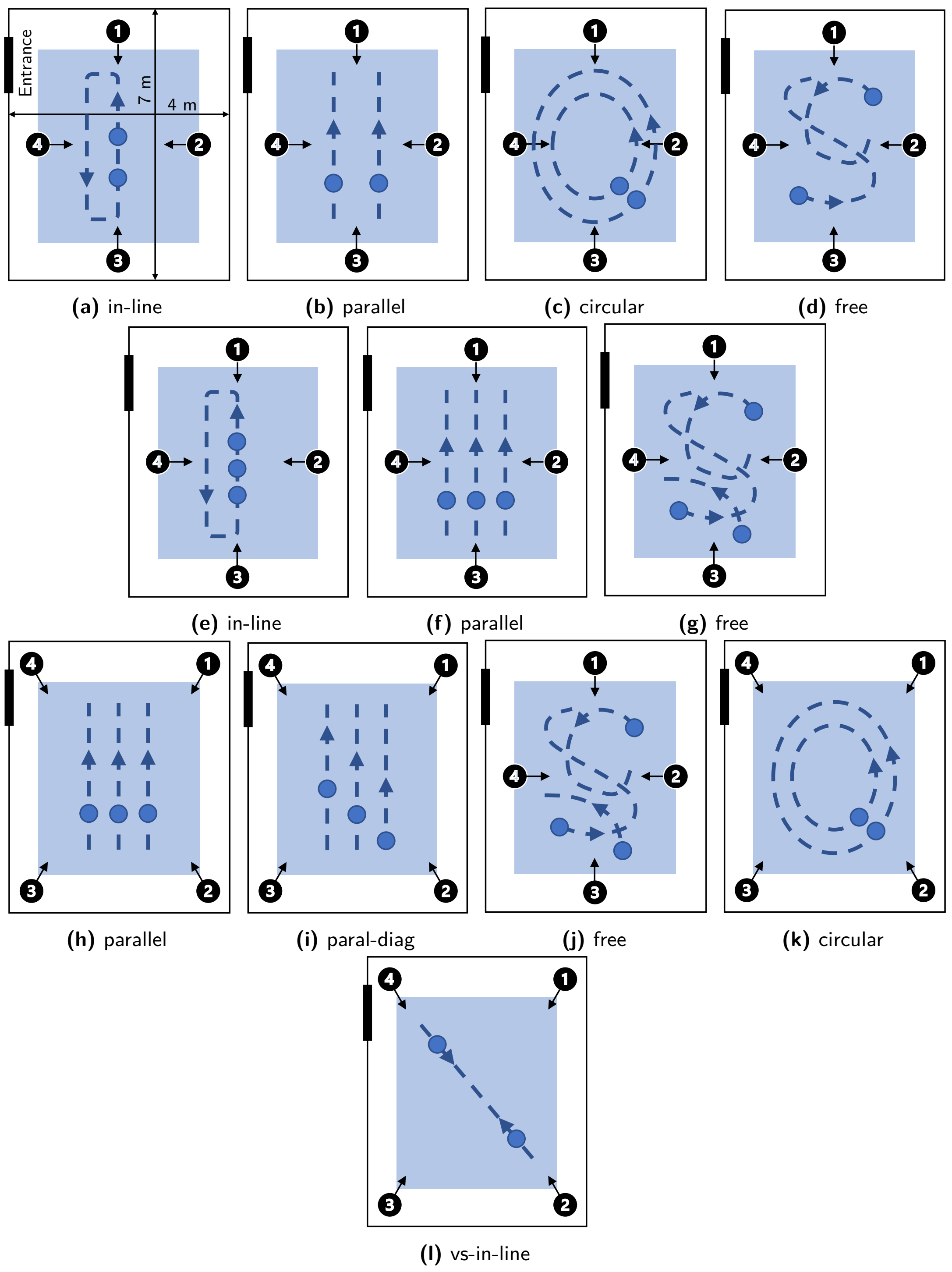


Figure 4.3: Setup schemes. The black numbered dots represent the RADAR devices and the arrows identify their pointing direction. The blue dots represent the moving people, while the dashed lines show the traveled trajectories in the direction given by the blue arrows. The first two rows show setup-1 deployments whereas the last two rows show setup-2 deployments.

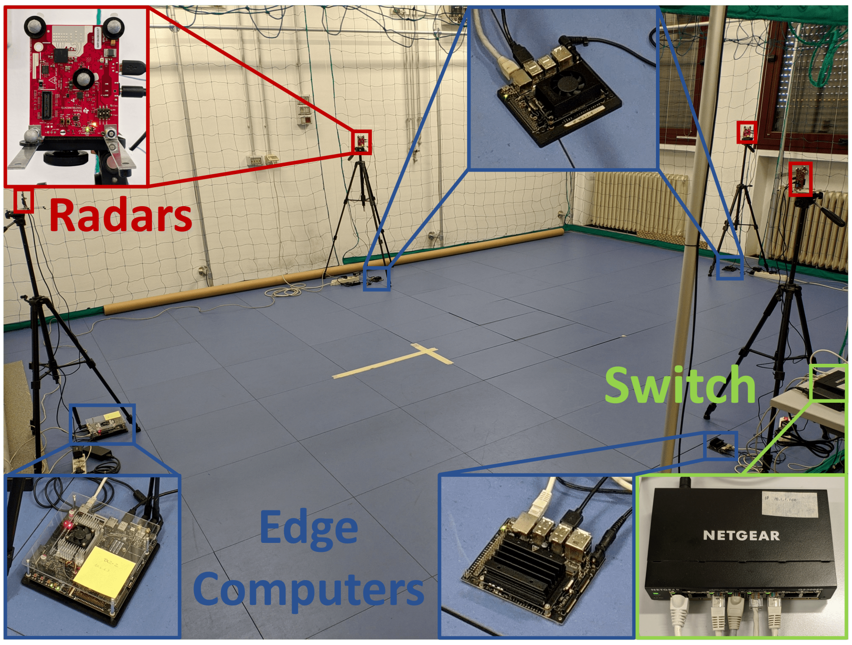


Figure 4.4: Experimental setup. The fusion center, not shown in the picture, is connected to the edge computers through a switch.

As a result of Eq. (4.16), the minimum and maximum eigenvalues of $\mathbf{P}_{\text{reg}}$ are $\lambda'_{\text{max}} + \delta$ and $\lambda'_{\text{min}} + \delta$, respectively. To limit the condition number of $\mathbf{P}_{\text{reg}}$, we specify an upper bound for its value, denoted by c^*. Then, such bound is enforced by computing δ from $\text{cond}(\mathbf{P}_{\text{reg}}) = (\lambda'_{\text{max}} + \delta)/(\lambda'_{\text{min}} + \delta) \le c^*$, which is solved with equality by

$$\delta = \max\left(0, \frac{\lambda'_{\text{max}} - c\lambda'_{\text{min}}}{c^* - 1}\right). \tag{4.17}$$

In our experiments, we empirically decided to adopt $c^* = 50$. ORACLE applies the correction in Eq. (4.15) whenever a covariance (or precision) matrix is non positive definite. The regularization in Eq. (4.16), instead, is used if the condition number of a covariance (or precision) matrix is above c^*.

4.5.2 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. 4.3a) equipped with a motion tracking system featuring 10 cameras. This provides the ground truth (GT) 3D localization of a set of markers placed on the RADARs and on the moving subjects with millimeter-level accuracy. We considered 2 different scenarios with 4 RADARs and 1, 2, and 3 moving targets. Fig. 4.3 shows the locations and orientations of the RADARs in the 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 6 possible different trajectories: *(i) in-line*, identifying a movement along a straight line, one subject after the other; *(ii) parallel*, identifying a movement along parallel lines; *(iii) circular*, corresponding to the two subjects following parallel and circular trajectories; *(iv) free*, where all subjects could move freely in the room; *(v) paral-diag*, identifying a movement on parallel lines but with the subjects spaced apart along the movement directions; and *(vi) vs-in-line*, where the two subjects moved one towards the other following the same linear trajectory. All trajectories are depicted in Fig. 4.3. In total, we collected 55 sequences, each 40 s long. In every sequence, subjects were tracked by all 4 RADARs simultaneously and independently. Then, tracking information has been fused considering all possible combinations of 1 to 4 RADARs. However, we experienced the Jetson Nano DevKit edge computer not being able to properly track more than 2 targets simultaneously, and we noticed the issue after the experiments. For this reason, in order to provide reliable results, we decided to show results with up to 3 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. These setups are represented in Fig. 4.11. For this part of the experiments, we collected 40 sequences with 2 subjects and 3 RADARs. Results pertaining to this part of the dataset will be presented in Section 4.5.7.

4.5.3 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 4.3.2. The *position error* is defined as the Euclidean distance between the estimated position of the RADAR and its true position. In order to assess the tracking performance, we adopt the *Multiple Object Tracking Performance Accuracy* (MOTA) metric, 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 [38].

4.5.4 Self-calibration

As previously mentioned, in the first part of this work we are presenting an enhanced version of mmSCALE [12], our self-calibration algorithm. The enhancement, consisting in the addition of the *masking* phase to the self-calibration procedure, allows to handle a wider range of cases, where the previous version was more prone to errors, and to achieve a general improvement in the

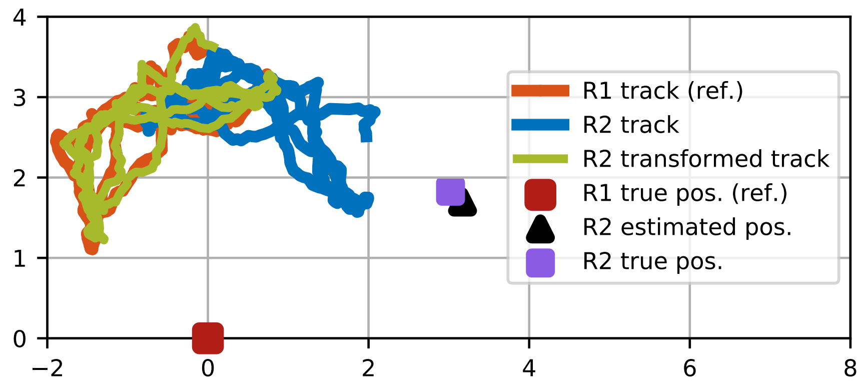


Figure 4.5: Example of self-calibration with a *free* trajectory.

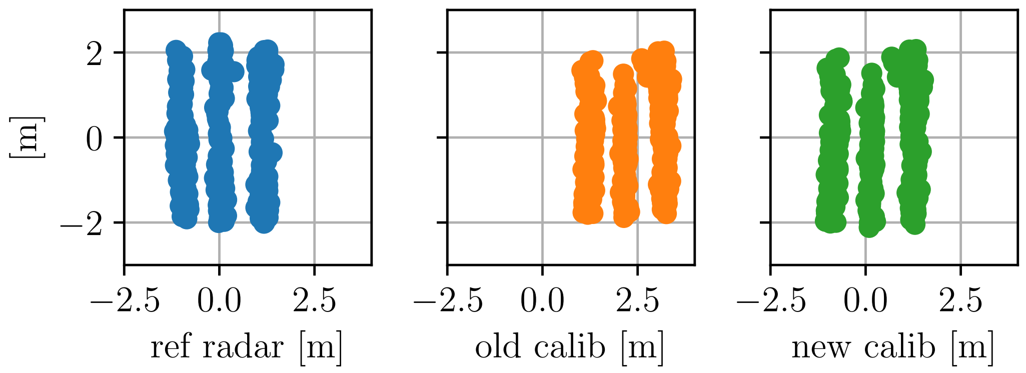


Figure 4.6: Example of a situation where the old algorithm fails while the new one doesn't. In this case, linear, similar trajectories led to a mistake in the track association phase, resulting in a shift in the estimated location of the RADAR.

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.

Qualitative results. Fig. 4.5 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), so as 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 4.4.1) is able to lead to a correct association for most situations. However, there are some particular cases where it is not sufficient. Fig. 4.6 shows an example of one of such cases with 3 subjects following a *parallel* trajectory. The blue lines represent the trajectories of the reference RADAR in its RS, the orange

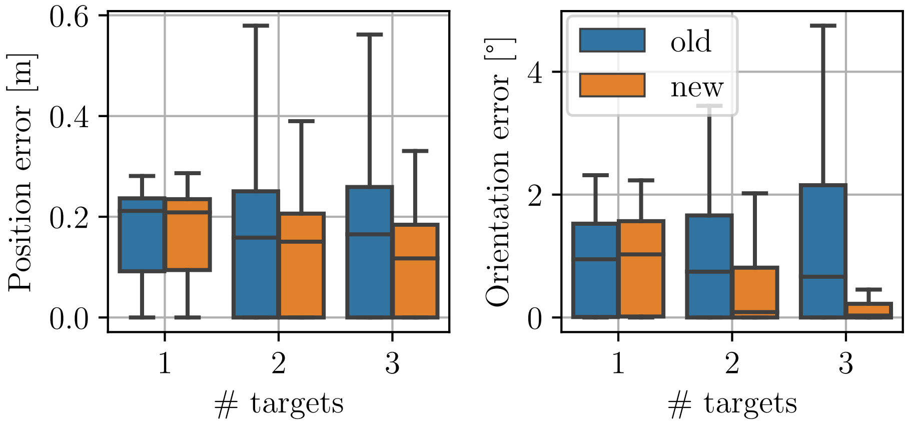


Figure 4.7: Comparison of the self-calibration results when using the old and the new self-calibration algorithm as a function of the number of targets tracked during the calibration phase.

lines shows 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.

Position and orientation errors. Fig. 4.7 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 is able to handle correctly. Tab. 4.1 shows the numerical results in terms of median and interquartile range (IQR). It also shows the difference between the new and the old algorithm.

4.5.5 Fusion center tracking accuracy

We evaluate the performance of the fusion algorithm in two cases: *(i)* using the roto-translation parameters obtained from the GT; and *(ii)* using their estimations from the self-calibration algorithm. In order 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. 4.8 shows the tracking results versus the number of targets tracked and the number or RADARs used for the fusion. In the figure, xT-SF

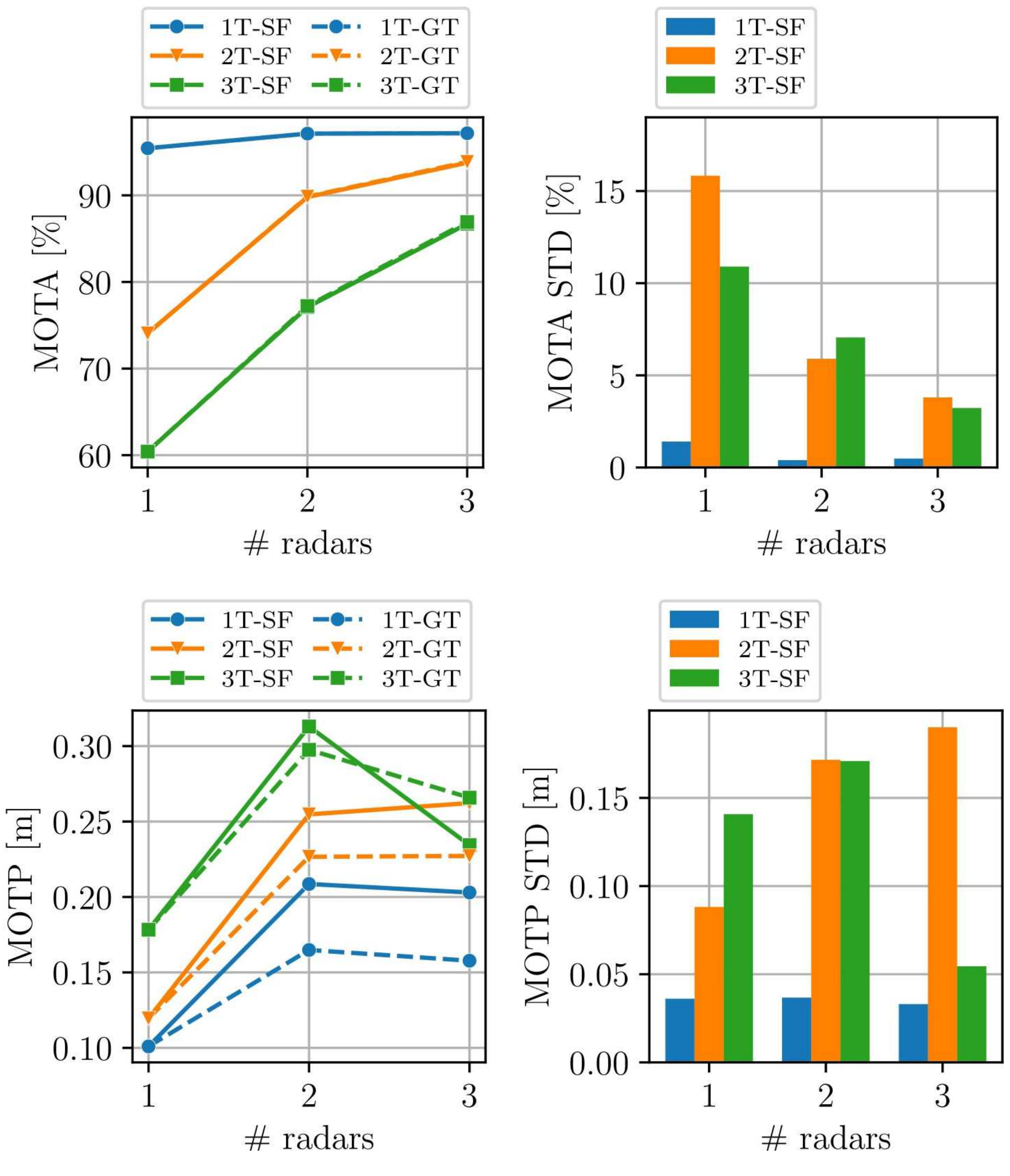


Figure 4.8: Average MOTA and MOTP as a function of the number of RADARs used for the fusion and of the number of targets tracked. Solid and dashed lines identify, respectively, results obtained using self-calibration (SF) and ground-truth (GT) to estimate RADARs' location and orientation.

Table 4.1: 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)

Table 4.2: 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

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 deviations for the self-calibration results. The values are computed as the average over all the sequences with the same number of RADARs and targets. Numerical results using self-calibration are presented in Tab. 4.2, 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 3 targets are tracked, leading to an improvement with respect to single sensors of +27%. This is due to the fact that 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 straightfowardly 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

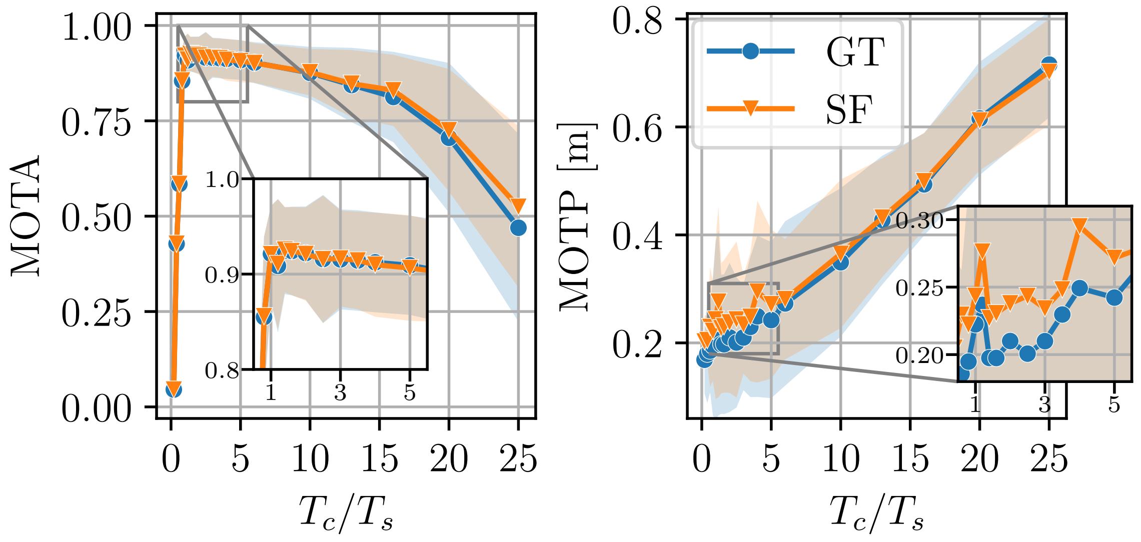


Figure 4.9: Average MOTA and MOTP versus different values of the ratio T_c/T_s when calibration is performed using GT measurements (GT), or with the self-calibration (SF). The shaded areas represent 1 standard deviation.

fused. Following the 3T lines, instead, MOTP increases from 1 to 2 RADARs and then decreases when 3 RADARs are used, reaching the same values of the 2T lines. A possible explanation for this is that 2 RADARs are not enough for tracking 3 targets, leading to errors in the track associations that cause the MOTP to increase. 3 RADARs, instead, have better tracking capabilities and can better handle 3 targets. For the most challenging scenario (3 targets), MOTP is 31 cm when 2 RADARs are used and 23 cm when 3 RADARs are used. MOTP is slightly lower when using GT rather than self-calibration because of the more precise knowledge about sensors' position and orientation.

As a final test, we acquired some sequences where we fuse all of the 4 RADARs for various 2-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 an occlusion-resilient people tracking through a self-calibrated RADAR network, requiring almost no human intervention.

4.5.6 Robustness to reduced fusion rate

In certain resource-constrained applications it may be useful to reduce the FC processing rate of the sensors data, in order 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.

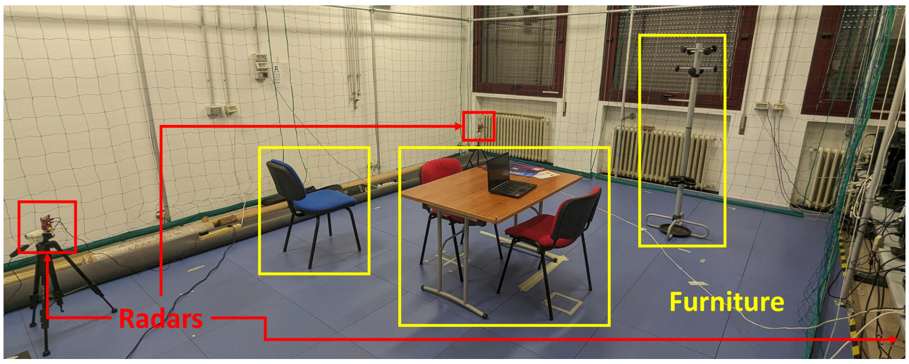


Figure 4.10: Picture of the setup with furniture.

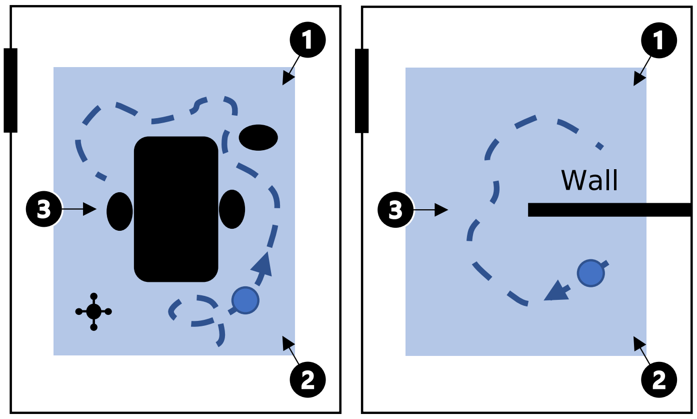
In Fig. 4.9, we show the MOTA and MOTP curves as a function of the ratio T_c/T_s. This is varied by fixing $T_s = 66.7$ ms (15 fps) and changing T_c from $0.2T_s$ to $25T_s$. Values are obtained by averaging all experiments with 3 RADARs. We can identify three regions.
(i) For $T_c/T_s < 0.8$, the MOTA 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.
(ii) For $0.8 \leq T_c/T_s \leq 5$, our system achieves the best performance in terms of MOTA, i.e., 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* 5 *times* with negligible performance degradation.
(iii) For $T_c/T_s > 5$, MOTA degrades slowly and MOTP increases. This is because the time-step of the FC, especially towards 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 *(i)*-*(iii)*. 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.

4.5.7 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.

Presence of furniture

The first scenario contains furniture, placed as shown in Fig. 4.11a. 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 Tab. 4.3, with one (1T) and two (2T) targets moving in the room. By comparing the obtained results with those in Tab. 4.2 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



(a) Furnture layout. **(b)** U-shaped corner.

Figure 4.11: Schemes of the setups with furniture (a) and simulating the U-shaped corner of a corridor (b).

-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 RADAR combine their data via the proposed data fusion algorithm.

U-shaped corner

In the second scenario, which we refer to as “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 block the mmWave signal. This replicates a U-shaped corner in which targets are entirely occluded for some of the RADAR sensors (see Fig. 4.11b), while they are always in the FoV of at least one sensor. Specifically, targets moving in the proximity of RADAR 1 in Fig. 4.11b cannot be detected by RADAR 2 and vice versa. Tab. 4.3 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 the large obstacle. This is a result of the combined view provided by the fusion center, 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 4.4.2. It follows that such procedure is robust to subjects that frequently enter and leave the field of view of RADAR sensors due to occlusions.

Table 4.3: Tracking performance with furniture (Furn-) and in a U-shaped corner.

		1R	**3R**
Furn-1T	**MOTA** [%]	75 ± 8	96 ± 2
	MOTP [m]	0.19 ± 0.02	0.16 ± 0.01
Furn-2T	**MOTA** [%]	65 ± 10	94 ± 3
	MOTP [m]	0.16 ± 0.03	0.16 ± 0.01
U-shaped corner	**MOTA** [%]	50 ± 7	94 ± 4
	MOTP [m]	0.17 ± 0.02	0.18 ± 0.03

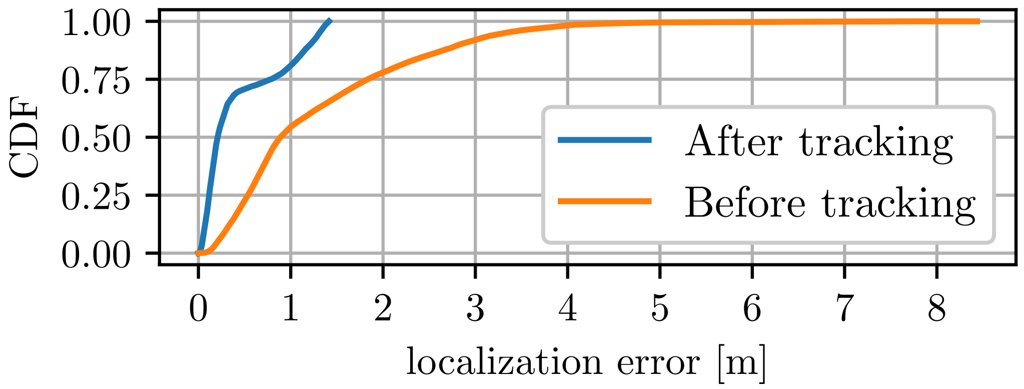


Figure 4.12: Empirical localization error CDF with and without using a tracking algorithm on the single sensors, thus relying only on the raw RADAR measurements.

4.5.8 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 to the different subjects is crucial. This is especially true for multiple targets scenarios, where there is ambiguity regarding which target generates which reflection. Fig. 4.12 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 enviroments, as they are subject to fluctuations, as well as missed and ghost detections. Note that the use of a (multi-sensor) tracking step sets ORACLE apart from other approaches, such as [41]: 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 fusion center.

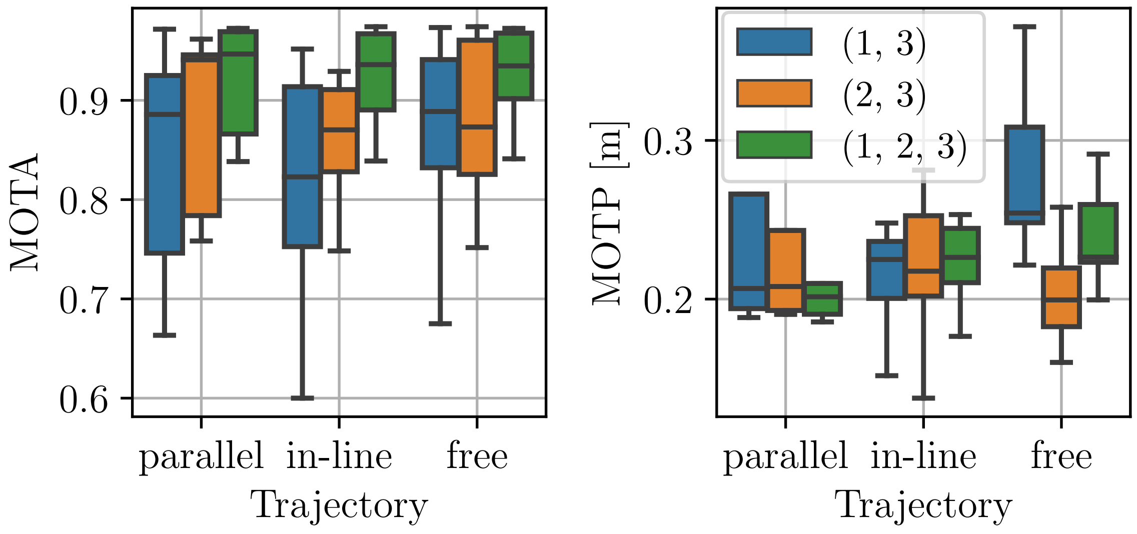


Figure 4.13: Box plots of MOTA and MOTP for particular trajectories, as a function of specific combinations of fused RADAR. More precisely, $(1, 3)$ has facing RADARs, $(2, 3)$ has perpendicular RADARs, $(1, 2, 3)$ fuses all of the three. Radar numbers correspond to those in Fig. 4.3, first row (setup-1).

4.5.9 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 work and would require a denser deployment 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. 4.13 we compare the results for 3 trajectories and some specific combinations of RADARs for the fusion. In particular, according to setup-1 (see Fig. 4.3, 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 3 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 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 *(i)* a larger number of RADARs is to be preferred over a lower one, and *(ii)* RADARs with more diverse points of view provide, in general, better tracking results.

4.5.10 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 at the RADAR

Table 4.4: Comparison with 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

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 4.3.1) with the well-known Multiple Hypothesis Tracking (MHT) scheme [55]. For MHT, we used the same inputs as for NN-JPDAF, that is, we applied DBSCAN [34] clustering to the RADAR point clouds and used the clusters centroids as detections. MHT was tested on a subset of the dataset including 2 targets and *free* and *parallel* trajectories, presenting the results in Tab. 4.4. 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.

4.5.11 Summary of the implementation challenges

In addition to the challenges posed by ORACLE's design, presented in Section 4.3, its experimental implementation and validation involves several other important aspects. The main ones are discussed in the following.

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 between the edge nodes and the fusion center. In ORACLE, this is solved by using the RadNet platform [13], which provides all the necessary functionalities.

Time synchronization between network nodes

As explained in Section 4.4, 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 4.5.2, we experienced the Jetson Nano DevKit edge computer not being able to timely track more than 2 targets simultaneously, due to its low computational power, introducing a delay in the data timestamps and, therefore, making them unusable.

4.6 Concluding remarks

In this chapter, 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 reference system. Then, it exploits such estimates to fuse the information about people tracked by different RADARs at a fusion center, 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 3 targets moving. ORACLE handles different time steps for the single sensors and for 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.

5 Contact Tracing and Temperature Screening via mmWave and Infrared Sensing

This chapter tackles the problem of fusing information from mmWave RADARs with measurements from a thermal camera (TC). In particular, it develops a real-time, integrated, radio and infrared sensing system to jointly perform unobtrusive elevated skin temperature screening and privacy preserving contact tracing in indoor environments.

5.1 Introduction

This chapter tackles the problem of designing a real-time, integrated radio and infrared sensing system to jointly perform unobtrusive elevated skin temperature screening and privacy preserving contact tracing in indoor environments.

Lately, *social distancing* has become a primary strategy to counteract the COVID-19 infection. Many research works [56], [57] have shown that it is an effective non-pharmacological approach and an important inhibitor for limiting the transmission of many contagious diseases such as H1N1, SARS, and COVID-19. Along with social distancing, *elevated skin temperature detection* and *contact tracing* have proven to be key to effectively contain the pandemic [58]. However, available methods to enforce these countermeasures often rely on RGB cameras and/or apps that need to be installed and continuously run on people's smartphones, often rising privacy concerns [59]. Moreover, currently adopted methods to screen people's temperature require individuals to stand in front of a thermal sensor, which may be impractical in heavily frequented public places.

Here, milliTRACE-IR, a joint mmWave RADAR and infrared imaging sensing system is de-

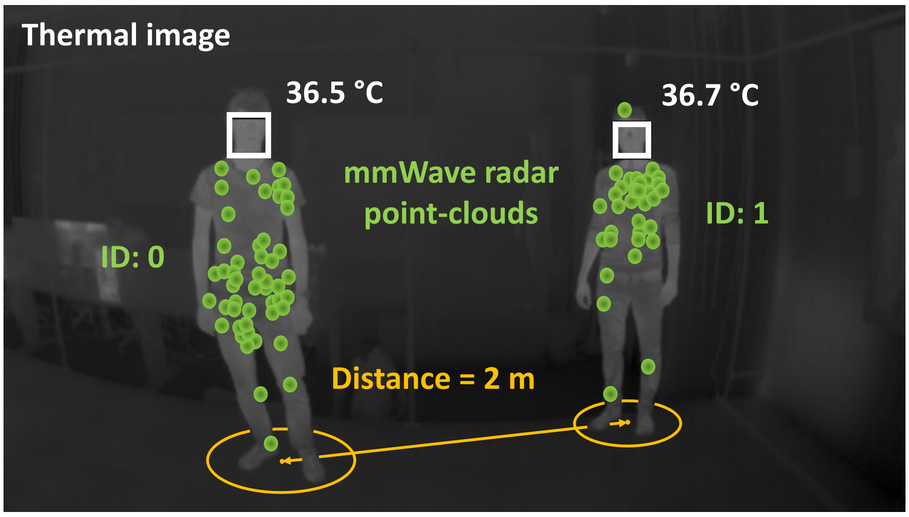


Figure 5.1: milliTRACE-IR performs body temperature screening and interpersonal distance estimation via sensor fusion of an infra-red thermal camera and mmWave RADARs. Individual gait features contained in the mmWave reflections enable contact tracing across different rooms.

signed and validated. milliTRACE-IR performs unobtrusive and privacy preserving human body temperature screening and contact tracing in indoor spaces (see Fig. 5.1). Next, its main components are discussed, emphasizing their novel aspects and the joint processing of the acquired sensor data.

mmWave RADAR: The RADAR analyzes the reflections of a transmitted mmWave signal off the individuals that move in the monitored environment, returning *sparse point-clouds* that carry information about the subjects' locations and the velocity of their body parts. A novel point-cloud clustering method is designed, combining Gaussian mixtures [60] and the density-based DBSCAN [34] algorithm, to distinguish the mmWave radio reflections from the subjects, as they move as close as 0.2 m to one another. The so obtained point-cloud clusters are used to track the subjects' positions in the physical space by means of a Kalman Filter (KF) [22], and to obtain their gait-related features through a deep-learning based feature extractor. Finally, a novel person re-identification algorithm is proposed by exploiting weighted extreme learning machines (WELM).

Thermal camera: The infrared imaging system, or TC, returns images whose pixels contain information on the *temperature of the objects* in the TC FoV. To measure the subjects' temperature, at first, YOLOv3 [61] is used to perform face detection in the TC images, by bounding those areas containing a human face. Hence, the obtained bounding boxes are tracked through an extended Kalman filter (EFK) [62] and the subjects' temperature is estimated by accumulating readings for each EFK track, according to a dedicated estimation and correction procedure. Through the EFK, the subject's distance from the TC is also estimated from the size of the corresponding bounding box by considering the non-linear part of the EFK, which is approximated by fitting a function over a set of experimental data points.

Radar and thermal camera data fusion: Tracks in the RADAR reference systems are associated with those in the TC image plane via an original algorithm that finds optimal matches for the readings taken by the two sensors, through their *joint* analysis. This makes it possible to take temperature measurements from a subject and reliably associate them with the highly precise tracking of his/her movement performed by the RADAR. In addition, the joint analysis of RADAR and TC data allows refining the temperature estimated through the TC: to mitigate the influence of the distance on the temperature readings [63], a regression function that provides temperature correction coefficients is fit from training data. The final temperatures are obtained using such function with the accurate distances retrieved from the RADAR.

Hence, once a subject's temperature is measured, it is associated with the corresponding RADAR track and the subject's movements and contacts inside the building are accurately monitored, by re-identifying the subject as he/she moves across the FoV of different RADAR devices. To the best of the author's knowledge, milliTRACE-IR is the first system that achieves temperature screening and human tracking through the joint analysis of RADAR and TC signals. Furthermore, it concurrently performs body temperature screening and contact tracing, while these aspects have been previously dealt with separately. A sensible usage model for the system is as follows: the TCs shall be deployed in strategic locations to allow an effective temperature screening, such as facing the building/room entrance, to ensure that people's faces are seen frontally for a reasonable amount of time, and that their TC images are only taken when they enter or leave the building/room. On the other hand, the RADAR can be utilized to track the subjects while moving inside the monitored indoor space. This ensures higher privacy with respect to RGB cameras.

The main contributions of the present work are:

1. milliTRACE-IR, a joint *mmWave RADAR and infrared imaging sensing system* that performs unobtrusive and privacy preserving human body temperature screening and contact tracing in indoor spaces is designed and validated through an extensive experimental campaign.

2. A novel *data association* method is put forward to robustly associate tracks obtained from the mmWave RADAR and from the TC, where the RADAR returns the people coordinates in the physical space and the TC identifies people's faces in the thermal image space. The achieved precision and recall in the associations are as high as 97%.

3. An original *clustering algorithm for mmWave point-clouds* is devised, making it possible to resolve the RADAR reflections from subjects as close as 0.2 m.

4. A new WELM based *person re-identification* procedure is presented. The WELM is trained at runtime on previously unseen subjects, achieving an accuracy of 95% over six subjects with only 3 minutes of training data.

5. A novel method is designed to perform *elevated skin temperature screening* as people move freely within the FoV of the TC, without requiring them to stop and stand in front of the thermal sensor. For this, a dedicated approach is presented to mitigate the distortion in

the TC temperature readings as a function of the distance, by also leveraging the accurate distance measures from the RADAR. Through this method, worst-case errors of 0.5 °C are obtained.

The chapter is organized as follows. In Section 5.2, the related work is discussed. Section 5.3 introduces some basic concepts about thermal imaging systems, while in Section 5.4 the proposed approach is thoroughly presented. Section 5.5 contains a description of the implementation of milliTRACE-IR and an in depth evaluation on a real experimentation setup. Concluding remarks are provided in Section 5.6.

5.2 Related Work

In the literature, almost no work has focused on a joint approach to social distancing and people's body temperature monitoring which preserves the privacy of the users. Here, several prior works in related areas are discussed, highlighting the differences with respect to the proposed system.
Social distancing monitoring: Social distancing has been one of the most widely employed countermeasures to contagious diseases outbreaks [56]. Real-time monitoring of the distance between people in workplaces or public buildings is key for risk assessment and to prevent the formation of crowds. Existing approaches use either wireless technology like Bluetooth or WiFi [64], [65], which require the users to carry a mobile device, or camera-based systems [66], which are privacy invasive. Other approaches use the received signal strength indication (RSSI) from cellular communication protocols [56] or wearables [67], although these are often inaccurate, especially when used in crowded places [56]. A lot of effort has been put into designing person detection and tracking algorithms for crowd monitoring and people counting [68] by using fixed surveillance cameras and mobile robots [69]. The main drawbacks of these methods are the intrinsic difficulty in estimating the distance between people from images or videos, along with the fact that the users have to be continuously filmed during their daily lives, which raises privacy concerns.

Concurrently, a large body of work has focused on ultra-wideband transmission for people tracking [4], [10], e.g., using mmWave RADARs, as these naturally allow measuring distances with decimeter-level accuracy. However, none of these works has tackled the problem of estimating interpersonal distances when people are very close to one another for extended periods of time; this is especially difficult with radio signals, as the separation of the reflections from different subjects becomes challenging.
Passive temperature screening: Infrared thermography is widely adopted for non-contact temperature screening of people in public places [70]. Due to the COVID-19 pandemic, there has been a growing interest in developing screening methods to measure the temperature of multiple subjects simultaneously, without requiring them to collaborate and/or to carry dedicated devices [71]. Approaches that involve the use of RGB cameras, e.g., [72], share the aforementioned privacy-related limitations.

The authors of [63] developed a Bayesian framework to measure the body temperature of multiple users using low-cost passive infrared sensors. The distance from the sensors and the

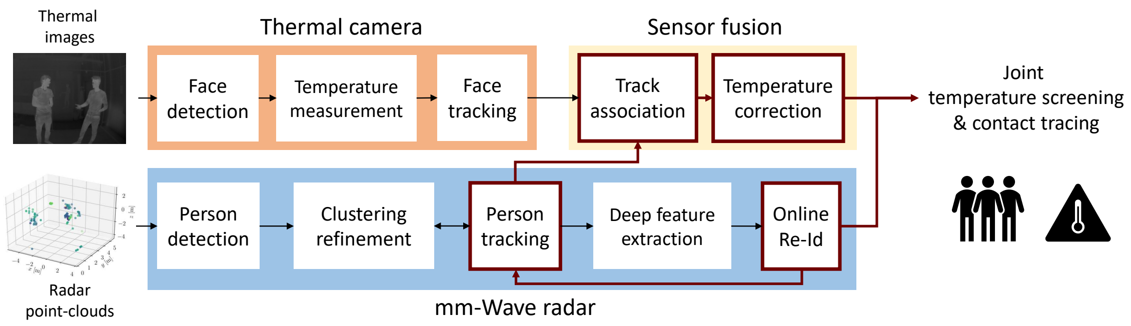


Figure 5.2: milliTRACE-IR signal processing workflow.

number of subjects is also obtained. However, the working range of this system is very short (around 1.5 m for precise temperature estimation), so it is deemed unapt for monitoring a large indoor area.

Radar-thermal imaging association and fusion: Sensor fusion between RADARs and RGB cameras has been extensively investigated, see, e.g., [73], [74], while the joint processing of mmWave RADAR data and infrared thermal images was marginally treated [7]. In addition, the last paper only deals with the detection of humans using thermal imaging and does not address body temperature screening.

The present work is focused on the *data association* between a thermal camera and a mmWave RADAR over short periods of time, using the accurate RADAR distance estimates to refine the temperature reading. This makes it possible to consider scenarios where the thermal camera only covers a small portion of the environment (e.g., the entrance) so as to preserve the subjects' privacy, while a mmWave RADAR network can effectively monitor the whole indoor space.

mmWave RADAR person re-identification (Re-Id): RF based person Re-Id is a recent research topic. So far, many works have focused on person identification [33], [75], where the subjects to identify have been previously seen by the system, typically via a preliminary training phase. Re-Id is more challenging, as it addresses the recognition of *unseen* subjects, for which only a few radio samples are collected during system operation. Differently from camera image based Re-Id methods [76], RF approaches need to profile the users across time intervals of a few seconds, to extract robust person specific features [77]. To the best of the author's knowledge, only two works have proposed solutions to this problems [77], [78]. In both cases, a deep learning method trained on a large set of users is used to extract features from the human gait. At test time, the features obtained from the subjects to be re-identified are compared against those of a set of known individuals using distance-based similarity scores. This approach entirely depends on the feature extraction process, and the classifier does not learn to refine its decisions at *runtime*, as new samples become available. This is a weakness, as the gait features extracted from mmWave RADARs are known to be variable, e.g., across different days [79]. Conversely, milliTRACE-IR combines deep feature extraction with fast classifiers which are continuously trained and refined as new data is collected; this improves the robustness of the identification task.

5.3 Preliminaries on Infrared Thermal Cameras

Infrared thermal imaging deals with detecting radiation in the long-infrared range of the electromagnetic spectrum ($\sim 8 - 15$ μm) and producing images of that radiation, called *thermograms*. According to the *Planck's Law*, infrared radiation is emitted by all objects with temperature $T > 0$ K [80]. Since the radiation energy emitted by an object is positively correlated to its temperature, from the analysis of the received radiation it is possible to measure the object's temperature.

A thermographic camera, or *thermal camera*, is a device that is capable of creating images of the detected infrared radiation. The operating principle is quite similar to that of a standard camera, and the same relations described by the so-called *pinhole camera model* hold [81]. Within this approximation, the coordinates of a point $\boldsymbol{a} = [a_x, a_y, a_z]^T$ in the three-dimensional space are projected onto the image plane of an ideal pinhole camera through a very small aperture. Mathematically, this operation is described as $\boldsymbol{a}^{\text{proj}} = \boldsymbol{\Psi}\boldsymbol{a}$, where $\boldsymbol{a}^{\text{proj}}$ is the projected point and $\boldsymbol{\Psi}$ is the intrinsic matrix of the camera that contains information about its focal lengths, pixel dimensions and position of the image plane. However, when dealing with a real thermal camera, this approximation may be insufficient and the *radial* and/or *tangential* distortions introduced by the use of a lens and by inaccuracies in the manufacturing process may additionally have to be accounted for. On the image plane, an array of infrared detectors is responsible for measuring the received radiation, which is sampled and quantized to produce a digital information. The pixels of the final image that is returned by a thermal camera contain information about the temperature of the corresponding body/object part, encoded into the pixel intensity.

5.4 Proposed Approach

This work considers the problem of monitoring an indoor environment covered by multiple mmWave RADAR sensors, which span over different rooms and corridors. A few infrared thermal cameras are placed at strategic locations to perform accurate temperature screening of the people in the indoor space without compromising their privacy, e.g., at the building's entrance.

From a high-level perspective, milliTRACE-IR performs the following operations.
(1) **Person detection and temperature measurement:** When people enter the monitored indoor space, the system concurrently performs face detection from the infrared images captured by the thermal camera and person detection using the mmWave RADAR point clouds.

1. From the TC images, a face detector is used to obtain bounding boxes enclosing the faces of the detected subjects, (Section 5.4.2). A measure of their body temperature is obtained from the intensity of the thermal image pixels in the bounding box, see Section 5.4.3. While milliTRACE-IR works independently of the specific face detector architecture used, in the implementation YOLOv3 is used [61].

2. Concurrently, RADAR signal processing is used to detect and group the point-clouds from different subjects and estimate their positions and a novel clustering algorithm based on

DBSCAN and Gaussian Mixture models is put forward to separate the contributions of closeby subjects (Section 5.4.4).

(2) **Radar-TC person tracking:** Kalman filtering (KF) is independently applied to the TC images and to the RADAR point-clouds to respectively track the subjects' movements within the thermal images and in the indoor Cartesian space. Standard KF-based tracking in the thermal image plane is here modified to achieve a coarse estimation of the distance of the subjects, based on the dimension of their face bounding box Section 5.4.2. In this phase, each subject track is associated with a unique numerical identifier.

(3) **Radar-TC track association:** As a subject exits the FoV of the TC, his/her body temperature is associated with the corresponding trajectory from the mmWave RADAR, by performing a track-to-track association between TC tracks and RADAR tracks. This association algorithm is based on the subjects' distances from the TC, and on the RADAR estimated positions of the subjects, projected onto the thermal image plane (Section 5.4.5). After the association, the temperature measurement is corrected accounting for the distance of each person from the TC, using the more precise distance estimates provided by the RADAR, Section 5.4.3.

(4) **Radar-based person re-identification:** During the RADAR tracking process, the point-cloud sequences generated by each subject are collected and fed to a deep neural network that performs gait feature extraction (Section 5.4.7). The resulting gait features are organized into a labeled training set, where labels are obtained from the track identifiers. When a subject exits the FoV of a RADAR and enters that of another RADAR placed in a different room or corridor, a weighted extreme learning machine (WELM) based classifier [82] is trained on-the-fly and used to re-identify the subject at runtime (Section 5.4.9). This robust and lightweight person Re-Id process, based on the gait features extracted from the RADAR point-clouds, enables contact tracing across large indoor environments.

5.4.1 Notation

The system operates at discrete time-steps, $k = 1, 2, \dots,$ each with fixed duration of Δ seconds, also referred to as *frame* in the following. Boldface, capital letters refer to matrices, e.g., $\boldsymbol{X}$, with elements X_{ij}, whereas boldface lowercase letters refer to vectors, e.g., $\boldsymbol{x}$. $\boldsymbol{X}^{-1}$ denotes the inverse of matrix $\boldsymbol{X}$, and $\boldsymbol{x}^T$ denotes the transpose of vector $\boldsymbol{x}$. $\boldsymbol{x}_k$ refers to vector $\boldsymbol{x}$ at time k, x_j refers to element j of $\boldsymbol{x}$ and $(\boldsymbol{x}_k)_j$ is element j of $\boldsymbol{x}_k$. $\mathcal{N}(\mu, \sigma^2)$ indicates a Gaussian random variable with mean μ and variance σ^2. Notation $||\boldsymbol{x}||_2$ indicates the Euclidean norm of vector $\boldsymbol{x}$, while $||\boldsymbol{x}||_{\boldsymbol{\Gamma}} = \sqrt{\boldsymbol{x}^T\boldsymbol{\Gamma}\boldsymbol{x}}$ denotes the norm induced by matrix $\boldsymbol{\Gamma}$. The diagonal matrix with elements $x_1, x_2, \dots, x_n$ is denoted by $\text{diag}\,[x_1, x_2, \dots, x_n]$. $|\mathcal{X}|$ indicates the cardinality of set $\mathcal{X}$ while $\log(\cdot)$ denotes the natural logarithm.

5.4.2 Thermal Camera: Face Detection and Tracking

The detection of the subjects in the thermal camera images is performed by means of a face detector that computes rectangular bounding boxes delimiting the faces of the people within the FoV. The bounding boxes are used to track the positions of the subjects in the subsequent instants and to identify a region of interest (ROI) from which the temperature of the targets is obtained. milliTRACE-IR is independent of the particular face detector used, provided that it outputs bounding boxes enclosing the faces of the subjects. In the implementation, YOLOv3 [61] is used due to its excellent performance in terms of accuracy and speed.

To track the faces of the subjects in the image plane, an extended Kalman filter (EFK) is employed [62]. Define the *state* vector of a target subject at time k, as $\boldsymbol{x}_k = [x_k^c, y_k^c, \dot{x}_k^c, \dot{y}_k^c, h_k, d_k, \dot{d}_k]^T$, where x_k^c, y_k^c are the true coordinates of the center of his/her face in the thermal image, $\dot{x}_k^c, \dot{y}_k^c$ its velocities along the vertical and horizontal directions, h_k is the true height of the bounding box enclosing the subject's face, d_k, the distance of the target from the camera in the physical space, and $\dot{d}_k$ its time derivative (rate of variation).

The observation vector obtained from the YOLOv3 face detector, denoted by $\boldsymbol{z}_k = [\tilde{x}_k^c, \tilde{y}_k^c, \tilde{h}_k]^T$, contains noisy measurements of the face position and height (represented by the height of the bounding box), which are distinguished from their true values by the superscript "$\tilde{}$". Denote the observation noise by vector $\boldsymbol{r}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$, with $\boldsymbol{R} = \text{diag}\left(\sigma^2_{\tilde{x}^c}, \sigma^2_{\tilde{y}^c}, \sigma^2_{\tilde{h}}\right)$, with diagonal elements representing the (constant) observation noise variances of $\tilde{x}_k^c$, $\tilde{y}_k^c$ and $\tilde{h}_k$, respectively. In the implementation $\sigma^2_{\tilde{x}^c} = \sigma^2_{\tilde{y}^c} = 0.01$ and $\sigma^2_{\tilde{h}} = 20$ are used.

The EFK state transition model is defined as $\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k)$, where $f(\cdot)$ is the transition function, connecting the system state at time k, $\boldsymbol{x}_k$, to that at time $k+1$, $\boldsymbol{x}_{k+1}$, and vector $\boldsymbol{u}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})$ represents the process noise. In the model used in this work, the process noise includes 4 independent components, representing two random accelerations of the bounding-box center coordinates, u_k^x, u_k^y, a random noise term for the bounding-box dimension, u_k^h, and a random acceleration for the subject's distance, u_k^d. Therefore, it can be written $\boldsymbol{u}_k = \left[u_k^x, u_k^y, u_k^h, u_k^d\right]^T$ with covariance matrix $\boldsymbol{Q} = \text{diag}\left[\sigma_x^2, \sigma_y^2, \sigma_h^2, \sigma_d^2\right]$. In the implementation, $\sigma_x^2 = \sigma_y^2 = \sigma_d^2 = 5$ and $\sigma_h = 5.148$ are used (see Section 5.4.2).

Assuming that the target moves according to a *constant velocity* (CV) model, from the state definition it follows that

$$f(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{bmatrix} x_k + \Delta \dot{x}_k + u_k^x \Delta^2/2 \\ y_k + \Delta \dot{y}_k + u_k^y \Delta^2/2 \\ \dot{x}_k + u_k^x \Delta \\ \dot{y}_k + u_k^y \Delta \\ g\left(d_k + \Delta \dot{d}_k + u_k^d \Delta^2/2\right) + u_k^h \\ d_k + \Delta \dot{d}_k + u_k^d \Delta^2/2 \\ \dot{d}_k + u_k^d \Delta \end{bmatrix}, \tag{5.1}$$

where the only non-linear term is function $g(\cdot)$, which relates the subject's distance extracted by the thermal camera to the height h_k of the bounding-box enclosing his/her face. The proposed approach consists in *(i)* obtaining an estimate for $g(\cdot)$ in an *offline* fashion using training data, and *(ii)* using such estimate in the EFK model. These two steps are detailed next.

Estimation of function $g(\cdot)$

Function $g(\cdot)$ maps the distance of the target from the thermal camera d_k, at time k, onto the corresponding height of the bounding box, h_k, as follows,

$$h_k = g(d_k) + u_k^h. \tag{5.2}$$

Using N_t training samples $\{h_i, d_i\}_{i=1}^{N_t}$ containing the true distances of the target, d_i, and the measured bounding box height, h_i, $g(\cdot)$ is obtained solving an *offline* non-linear least-squares (LS) problem of the form

$$\underset{g}{\operatorname{argmin}} \sum_{i=1}^{N_t} (h_i - g(d_i))^2 . \tag{5.3}$$

From the equations of the pinhole camera model [81], $g(\cdot)$ is restricted to the family of hyperbolic functions with shape $g(d_i) = b_0/(d_i + b_1) + b_2$, reducing the problem to that of estimating the parameters b_0, b_1, and b_2, i.e.,

$$\underset{b_0,b_1,b_2}{\operatorname{argmin}} \sum_{i=1}^{N_t} \left(h_i - \frac{b_0}{d_i + b_1} + b_2 \right)^2 . \tag{5.4}$$

This optimization problem is here solved using the Levenberg-Marquardt algorithm [83] for non-linear LS fitting: with the experimental setup used in this chapter, $b_0 = 162.04, b_1 = 0.61, b_2 = -14.79$ are obtained.

Note that the process noise acts on the bounding-box dimension in two ways, inside the function $g(\cdot)$, modeling the uncertainty in the subject's distance due to the random acceleration, and through the additive term u_k^h, modeling the imperfect estimation of $g(\cdot)$ itself. The variance of u_k^h can be estimated from the residuals, after fitting the training measurements with function $g(\cdot)$.

Using $g(\cdot)$ in the EFK

Due to the non-linear dependence of the state $\boldsymbol{x}_k$ on the process noise $\boldsymbol{u}_k$, in the EFK operations the following transformed process noise covariance matrix is used [84]

$$\boldsymbol{Q}'_k = \boldsymbol{L}_k \boldsymbol{Q} \boldsymbol{L}_k^T, \quad \text{with } \boldsymbol{L}_k = \left. \frac{\partial f(\boldsymbol{x}_k, \boldsymbol{u}_k)}{\partial \boldsymbol{u}_k} \right|_{\hat{\boldsymbol{x}}_{k|k}}, \tag{5.5}$$

where matrix $\boldsymbol{L}_k$ is the Jacobian of function $f(\cdot)$ with respect to the process noise vector, evaluated for the current state estimate. Using the above system model, the system state estimate at time k, $\hat{\boldsymbol{x}}_k$, is recursively obtained along with the corresponding error covariance matrix, $\boldsymbol{P}_k$. By definition of the EFK state, this allows us to get a coarse estimate of the distance of the subjects from the TC, which is exploited in the RADAR-TC data association step, see Section 5.4.5.

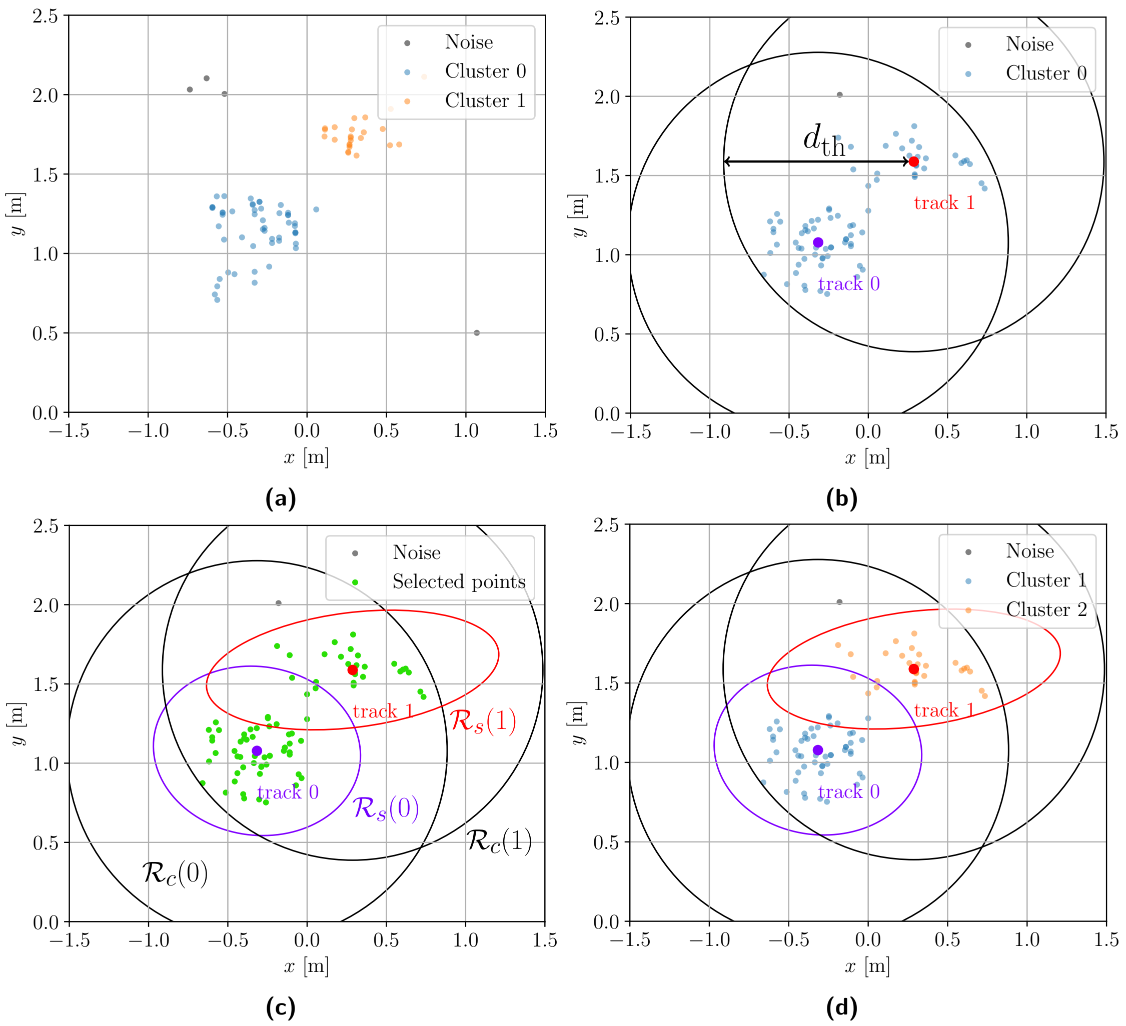


Figure 5.3: Illustration of the proposed clustering method. In (a) the point-clouds belonging to 2 subjects are well separated and DBSCAN outputs the correct clustering. In the next time-step, (b), DBSCAN fails and merges the two clusters into one. The proposed method selects the points to re-cluster using the tracks positions together with Eq. (5.6) and Eq. (5.7), as shown in (c), and outputs the correct result using Gaussian mixture (GM) on the selected points with $n_{\mathcal{G}} = 2$, see (d).

5.4.3 Thermal Camera: Subject Temperature Estimation

The body temperature is obtained from the thermal camera readings in the bounding-boxes contained in $\hat{\boldsymbol{x}}_k$, for each subject, and for all the time steps in which he/she is tracked by the EFK. At any given time k, a single (noisy) temperature measurement, $\tilde{T}_k$, is extracted by taking the maximum value across all the pixels in the current bounding box. Denoting by B_k the 2-D region of the image enclosed by the bounding box, and by B_{ki} the intensity of its pixel i, it holds $\tilde{T}_k = \max_i B_{ki}$.

5.4.4 mmWave Radar: Highly Accurate Clustering

milliTRACE-IR's tracking pipeline is based on the description given in Section 1.2.2. However, regarding the detection phase, DBSCAN has proven to be robust and accurate as long as the subjects do not come too close to one another [4], [33], [35], see also Fig. 5.3a. When this occurs (Fig. 5.3b), the algorithm often fails to distinguish between adjacent subjects, merging their contributions into a single cluster [85].

As a possible solution to DBSCAN drawbacks, one may adjust the parameters ε and m_{pts} so as to correctly resolve the clustering ambiguity, even for closely spaced targets. However, ε and m_{pts} interact in a complex and often unpredictable way, making the design of such adaptation rule difficult.

milliTRACE-IR adopts a different approach, which combines *(i)* the standard DBSCAN algorithm with fixed ε and m_{pts}, *(ii)* the spatial locations of the subjects, available from the tracking procedure, and *(iii)* the GM clustering algorithm [60]. The designed algorithm, reported in Alg. 5.1 and exemplified in Fig. 5.3, proceeds as follows. At first, the DBSCAN algorithm is applied to obtain an estimate of the clusters and a reasonable separation between the noise points and those belonging to actual subjects, using $\varepsilon = 0.4$ m and $m_{\text{pts}} = 10$. DBSCAN outputs a cluster label for each point $\boldsymbol{p} \in \mathcal{P}_k$, denoted by $\ell_{\boldsymbol{p}}$. Clusters are denoted by $\mathcal{C}_n$, and their centroids by $\bar{\boldsymbol{c}}_n$, with $n = 1, \dots, n_k$.

The next step is to identify which of the tracked subjects get closer than a critical distance d_{th} from one another. The clusters provided by standard DBSCAN for these subjects are expected to be incorrect, as the point-cloud data from these would be merged into a single cluster. To pinpoint these subjects, their KF state is leveraged, which corresponds to a filtered representation of their trajectories. Consider track t at time k, its coordinates are predicted as $\hat{\boldsymbol{s}}_k^t = \boldsymbol{A}\hat{\boldsymbol{s}}_{k-1}^t$ (see line $2-3$ in Alg. 5.1). For any two subjects with associated tracks t and t', milliTRACE-IR checks whether $||\hat{\boldsymbol{s}}_k^t - \hat{\boldsymbol{s}}_k^{t'}||_2 < d_{\text{th}}$. If this occurs, as shown in the example of Fig. 5.3b for tracks $t = 0$ and $t' = 1$, t and t' are termed *nearby subjects*. Hence, define $\mathcal{G}$ as the set of subjects that are mutually within a radius of d_{th} from one another. A group $\mathcal{G}$ can be constructed starting from any subject and recursively adding all the subjects who are closer than d_{th} from any of the set members. If a subject has no other subjects within distance d_{th}, it will be the only member of his group. Collecting all the disjoint groups, constructed from the maintained tracks at time k, set $\boldsymbol{\mathcal{G}}_k(d_{\text{th}})$ is obtained, containing all the nearby subjects groups. Once the nearby groups are identified, the ambiguities inside each group $\mathcal{G}$ containing more than one member are resolved by recomputing the clustering labels as follows. Consider a single group $\mathcal{G}$. To delimit the region where the clustering has to be refined, the following additional regions are defined. The sample covariance matrix of the *last* cluster associated with track t is denoted by $\boldsymbol{\Sigma}_n^t$, and contains information about the shape of the subject's cluster. The regions of the plane containing the points that are within a radius of d_{th} from $\hat{\boldsymbol{s}}_k^t$, can be written as

$$\mathcal{R}_c(t) = \left\{ \boldsymbol{x} \in \mathbb{R}^2 \text{ s.t. } \left|\left|\boldsymbol{x} - \hat{\boldsymbol{s}}_k^t\right|\right|_2 < d_{\text{th}} \right\}, \tag{5.6}$$

Algorithm 5.1 Clustering refinement method.

```
Require: States of the targets at time $k-1$, observed point-cloud at time $k$, $\mathcal{P}_k$.
Ensure: Labels $\ell_{\boldsymbol{p}}$, $\forall \boldsymbol{p} \in \mathcal{P}_k$.
 1: $\{\ell_{\boldsymbol{p}}\}_{\boldsymbol{p}\in\mathcal{P}_k}, \{\mathcal{C}_n\}_{n=1}^{n_k} \leftarrow$ DBSCAN$(\varepsilon, m_{\rm pts}, \mathcal{P}_k)$
 2: $\hat{\boldsymbol{s}}_k^t \leftarrow \boldsymbol{A}\hat{\boldsymbol{s}}_{k-1}^t$ all maintained tracks $t$
 3: Find groups of nearby subjects $\boldsymbol{\mathcal{G}}_k(d_{\rm th})$
 4: for each $\mathcal{G} \in \boldsymbol{\mathcal{G}}_k(d_{\rm th})$
 5:     $n_{\mathcal{G}} \leftarrow |\mathcal{G}|$
 6:     if $n_{\mathcal{G}} > 1$
 7:         $\mathcal{R}(\mathcal{G}) \leftarrow \bigcup_{t\in\mathcal{G}}(\mathcal{R}_c(t) \cap \mathcal{R}_s(t))$
 8:         $\mathcal{S} \leftarrow \{\boldsymbol{p} \in \mathcal{C}_n$ such that $\bar{\boldsymbol{c}}_n \in \mathcal{R}(\mathcal{G})\}$
 9:         discard $\ell_{\boldsymbol{p}}, \forall \boldsymbol{p} \in \mathcal{S}$
10:         $\{\ell_{\boldsymbol{p}}\}_{\boldsymbol{p}\in\mathcal{S}}, \{\pi_q\}_{q=1}^{n_{\mathcal{G}}} \leftarrow$ GM$(n_{\mathcal{G}}, \mathcal{S})$
11:         discard cluster $q$ if $\pi_q < \pi_{\rm thr}$
12:     end if
13: end for
```

and the regions of points with a squared Mahalanobis distance smaller than γ are

$$\mathcal{R}_s(t) = \left\{\boldsymbol{x} \in \mathbb{R}^2 \text{ s.t. } \left|\left|\boldsymbol{x} - \hat{\boldsymbol{s}}_k^t\right|\right|^2_{(\boldsymbol{\Sigma}_n^t)^{-1}} < \gamma\right\}. \tag{5.7}$$

In the implementation, $d_{\rm th} = 1.2$ m and $\gamma = 9.21$ were used.* Then, the labels assigned by DBSCAN to all the points belonging to a cluster whose centroid falls inside region $\mathcal{R}(\mathcal{G}) = \cup_{t\in\mathcal{G}}(\mathcal{R}_c(t) \cap \mathcal{R}_s(t))$, are discarded (lines 7 – 9 in Alg. 5.1).† This set of points is denoted by $\mathcal{S}$.

Then, the GM algorithm is applied to the points belonging to set $\mathcal{S}$ to refine the clusters within this region, see the green points in Fig. 5.3c. As GM requires the number of clusters to be specified in advance, it is set to be equal to the number of subjects in the group, i.e., $n_{\mathcal{G}} = |\mathcal{G}|$. The GM algorithm outputs the labels $\ell_{\boldsymbol{p}}$ for each point $\boldsymbol{p} \in \mathcal{S}$ and the weight of the Gaussian component associated with each GM cluster, $\pi_q \in [0, 1], q = 1, \dots, n_{\mathcal{G}}$, with $\sum_q \pi_q = 1$. The new labels are used to replace the ones previously found by DBSCAN (Fig. 5.3d), unless the GM clusters have very small weights, i.e., the new clusters having $\pi_q < \pi_{\rm thr}$ are discarded and treated as noise points. The threshold value used in the implementation is $\pi_{\rm thr} = 0.1/n_{\mathcal{G}}$.

The proposed method effectively solves the problem faced by DBSCAN in resolving subjects close to one another. The cost of this improvement is that an additional GM algorithm has to be applied to a subset of the point-cloud, however, at each time k the number of points in this subset is typically much smaller than that in the full point-cloud $\mathcal{P}_k$.

5.4.5 Radar and Thermal Camera Data Association

Upon tracking the subjects in the TC image plane and in the physical space, respectively using the measurements from the TC and from the mmWave RADAR sensor, a track-to-track association method is applied to link the movement trajectory of each person to his/her body temperature.

*The value of γ corresponds to a probability of 99% of falling inside the region, assuming that the points in the cluster are distributed on the plane according to a Gaussian distribution around $\hat{\boldsymbol{s}}_k^t$.

†Discarding a label corresponds to setting it equal to that used by DBSCAN to represent noise points.

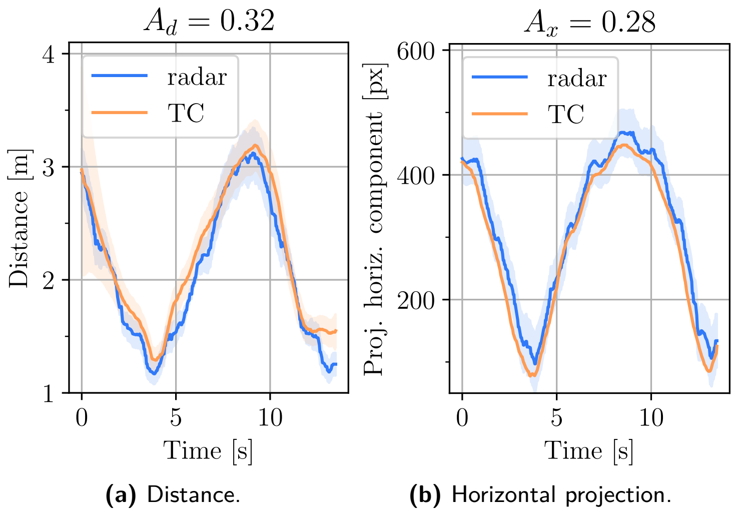


Figure 5.4: Example of distance (a) and horizontal projection (b) estimates from a track. The shaded areas represent the standard deviations. The corresponding values for A_d and A_x are shown above.

Assume that, at time k, the system has access to N_k^{rad} tracks from the RADAR sensor and N_k^{tc} tracks from the thermal camera, indicized by i and j, respectively. The data association strategy used in milliTRACE-IR consists in *(i)* computing a *cost* for each association $(i \leftrightarrow j)$, and *(ii)* solving the resulting combinatorial cost minimization problem to associate the best matching track pairs. The main challenge in the association of RADAR and thermal camera tracks is the design of a cost function that grants robustness in the presence of multiple targets, which may enter the monitored area in unpredictable ways. The key point is to gauge the similarity of the tracks by comparing them in terms of common quantities, which can be estimated from both devices.

Assume also that the two sensors are located in the same position and with the same orientation (co-located). In this setup, *(i)* the *distance* between the subjects and the sensors is the same, so its estimate should match for tracks representing the same subjects, and *(ii)* the RADAR KF states containing the coordinates of the subjects' positions can be projected onto the TC image plane; after this operation, the horizontal component of the RADAR projections and the horizontal component of the TC bounding boxes position should match for correctly associated tracks. To reliably associate RADARd and TC tracks, milliTRACE-IR uses a cost function consisting of the following components, see also Fig. 5.5

Estimated distance cost. Denote by d_k^i the estimated distance of RADAR track i, and by d_k^j the estimated distance of TC track j. Recalling that $\hat{\boldsymbol{s}}_k^i$ is the position of subject i at time k, d_k^i is computed using Pythagora's formula as $d_k^i = \sqrt{\left(\hat{\boldsymbol{s}}_k^i\right)_1^2 + \left(\hat{\boldsymbol{s}}_k^i\right)_2^2}$. Distance d_k^j, instead, is retrieved directly from the tracking state of the TC. Considering K subsequent time steps where RADAR track i and TC track j are both available, the estimated distance cost is defined as

$$A_d(i,j) = \frac{1}{K}\sum_{k=1}^{K}\frac{(d_k^i - d_k^j)^2}{\sigma^2_{d_k^i} + \sigma^2_{d_k^j}}, \tag{5.8}$$

where $\sigma^2_{d_k^i}$ and $\sigma^2_{d_k^j}$ represent the variances of the two distance estimates. An illustrative example is shown in Fig. 5.4a.

Projected horizontal component cost. The horizontal components of the RADAR state projection and of the TC bounding box center are respectively denoted by x_k^i and x_k^j. The RADAR positions provided by the KF state have only two dimensions, x and y (the first and second components of the state vector). However, three-dimensional vectors are needed for their proper projection onto the TC image plane. For this reason, a 0-valued z component is artificially added, under the assumption that the subjects' position at height 0 is the one being tracked. For this, an augmented subject's position vector, $\boldsymbol{a}_k^i = [\left(\hat{\boldsymbol{s}}_k^i\right)_1, \left(\hat{\boldsymbol{s}}_k^i\right)_2, 0]^T$, is defined. x_k^i is computed by projecting the RADAR coordinates $\boldsymbol{a}_k^i$ onto the TC image plane, as $\boldsymbol{a}_k^{i,\text{proj}} = \boldsymbol{\Psi}\boldsymbol{a}_k^i$ (see Section 5.3), applying to it a radial distortion based on the estimated distortion coefficients and retaining only the x-axis component. Projection x_k^j corresponds to the x coordinate of the TC tracked state. The projected horizontal component cost is defined, for K subsequent time steps of RADAR track i and TC track j, as

$$A_x(i,j) = \frac{1}{K}\sum_{k=1}^{K}\frac{(x_k^i - x_k^j)^2}{\sigma^2_{x_k^i} + \sigma^2_{x_k^j}}, \tag{5.9}$$

where $\sigma^2_{x_k^i}$ and $\sigma^2_{x_k^j}$ are the variances of the two estimates. An illustrative example is shown in Fig. 5.4b.

Track length coefficient. Recalling that Δ is the (constant) sampling interval, the proposed cost function accounts for the length K of the tracks that are to be associated, favoring longer tracks. To this aim, the following coefficient is defined,

$$\rho(K) = \frac{1}{\log(K\Delta)}. \tag{5.10}$$

Note that $\rho(K)$ is a weight factor for a cost (see the later Eq. (5.11)), which decreases with the track length K. This means that a smaller cost is implied when the associated tracks i and j are longer. Also, in the implementation, it holds $K > 1/\Delta$, so $\rho(K)$ is always positive.

Association cost function for RADAR and TC tracks. The association cost $A(i,j)$ for the tracks pair (i,j) (i refers to a RADAR track and j to a TC track) is obtained summing Eq. (5.8) and Eq. (5.9), to gauge how well the two tracks match in terms of their estimated distance across time, and estimated position on the horizontal projected axis on the TC image plane, respectively. The sum is then weighted by the coefficient of Eq. (5.10). Formally, $A(i,j)$ is given by

$$A(i,j) = \rho(K)\left[A_d(i,j) + A_x(i,j)\right]. \tag{5.11}$$

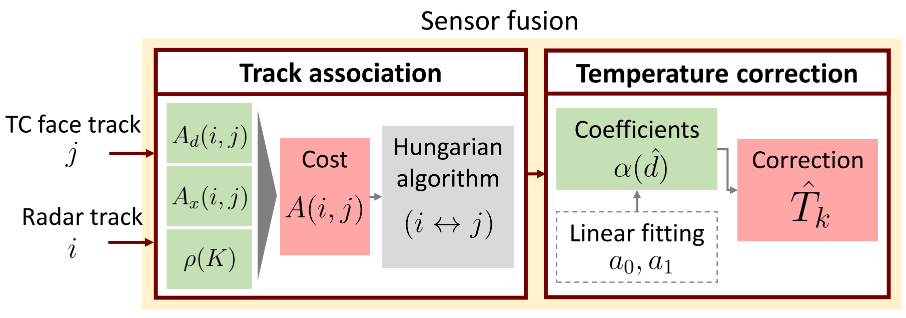


Figure 5.5: Block diagram of the sensor fusion step.

Costs $A(i,j)$, $i = 1, \dots, N_k^{\text{rad}}$, $j = 1, \dots, N_k^{\text{tc}}$, are arranged into an $N_k^{\text{rad}} \times N_k^{\text{tc}}$ matrix, and the optimal association of tracks is obtained by minimizing the overall cost, computed through the Hungarian algorithm [24]. The Hungarian algorithm takes the cost matrix as input and solves the problem of pairing each RADAR track with a single TC track (by minimizing the total cost), with an overall complexity of $O((N_k^{\text{rad}} N_k^{\text{tc}})^3)$.
In general, the RADAR and the TC would be deployed at different spatial locations. However, knowing their relative position and orientation, a roto-translation matrix $\mathbf{\Phi}$ can be obtained to geometrically transform the data into a new coordinate system where the TC and the RADAR sensors are co-located, as described above. In this work, the TC position and orientation are selected as the reference coordinate system, and the positions estimated from the RADAR sensor are transformed into it.

5.4.6 Temperature Correction

In line with [63], the direct reading of each subject's temperature , $\tilde{T}_k$, is subject to a scaling factor, $\alpha(d_k)$, with respect to the true temperature T, where $\alpha(d_k)$ depends on the distance from the TC, i.e.,

$$T = \alpha(d_k)\tilde{T}_k. \tag{5.12}$$

For an accurate temperature screening, the scaling factor $\alpha(d_k)$ is estimated from the training data, considering a linear model of the form

$$\alpha(d_k) = a_0 + a_1 d_k. \tag{5.13}$$

Using $N_t^{'}$ training measurements $\{\tilde{T}_i, d_i, T\}_{i=1}^{N_t^{'}}$, the fitting coefficients a_0, a_1 are obtained by solving

$$\underset{a_0,a_1}{\text{argmin}} \sum_{i=1}^{N_t^{'}} \left(T - \alpha(d_i)\tilde{T}_i\right)^2. \tag{5.14}$$

From the above optimization problem, in this work the above parameters are set to $a_0 = 1.116$, $a_1 = 0.013$. At system operation time, denoting by M the number of time-steps for which the

Table 5.1: Summary of the architecture and training parameters of the neural network (NN) used for gait feature extraction.

Architecture	
Layer/block	**Size**
PC features [79]	$3+2$ shared MLPs, (98, 196)
Temporal conv. [79]	3 Conv. (3×3), $32, 64, 128$ filt.
Temporal conv. [79]	3 Conv. (3×3), $256, 128, 32$ filt.
Global average pooling	32
Fully connected	32
L_2 normalization	32
Fully connected	16
Training parameters	
Learning rate	10^{-4}
Optimizer	Adam [86]
Number of epochs	250
$\mathcal{L}_{\text{cen}}$ weight, ω	0.5
L_2-regularization parameter	8×10^{-5}
Dropout rate	0.4
Triplet margin, μ	1

subject is correctly tracked by the EFK, his/her true temperature at time k is finally estimated as

$$\hat{T}_k = \frac{1}{M} \sum_{j=k-M+1}^{k} \alpha(\hat{d}_j)\tilde{T}_j, \tag{5.15}$$

where $\alpha(\cdot)$ is defined in Eq. (5.13), using the parameters obtained from Eq. (5.14), while $\hat{d}_j$ is an estimate of the distance obtained by the system at time-step j. To improve the temperature estimates, milliTRACE-IR performs sensor fusion by exploiting the association between the TC face tracks and the mmWave RADAR tracks (see Section 5.4.5). In Eq. (5.15), the coefficients $\alpha(\hat{d}_j)$ are computed using the distances estimated by the mmWave RADAR device, as these are much more accurate than those obtained from the TC. The impact of combining the temperature information from the TC and the accurate distance estimation capabilities of the RADAR is investigated in Section 5.5.3. The block diagram for the temperature correction step is shown in Fig. 5.5.

5.4.7 Extraction of Feature Vectors from mmWave Point-Clouds

To extract the gait features of the subjects, the NN proposed in [79], which was originally developed for person identification, is here adapted. The network uses a point-cloud feature extraction block inspired by PointNet [87], and followed by temporal dilated convolutions [88] to capture features related to the movement evolution in time. The proposed NN takes as input a RADAR point-cloud sequence, denoted by $\boldsymbol{Z}$, and outputs the corresponding feature vector $\boldsymbol{v} = \mathcal{F}(\boldsymbol{Z})$. Fig. 5.6 shows the block diagram of the NN. First, the network is expanded with respect to [79], using augmented point-cloud feature extraction blocks composed of 3 shared multi-layer perceptrons (MLPs) of

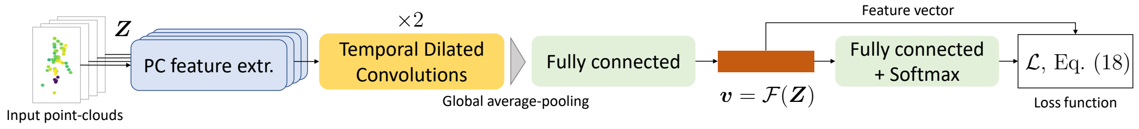


Figure 5.6: Block diagram of the NN feature extractor.

size 98 and 2 MLPs of size 196, yielding point cloud features of size 196×1. Then, 2 temporal convolution blocks are used, containing 3, 3×3, convolutional layers each, with $(32, 64, 128)$ and $(256, 128, 32)$ filters, respectively, for the two blocks, and dilation rates of $1, 2, 4$ for the 3 layers in each block. Then, after applying the same global average pooling operation of [79], a fully connected layer [86] is introduced before the classification output, which produces a vector $\tilde{\boldsymbol{v}}$ of dimension 32. The final feature vector is obtained using L_2-normalization on $\tilde{\boldsymbol{v}}$, i.e., $\boldsymbol{v} = \tilde{\boldsymbol{v}}/||\tilde{\boldsymbol{v}}||_2$. A summary of the NN layers and their parameters is provided in Tab. 5.1.

Training

The NN is trained to produce representative feature vectors, $\boldsymbol{v}$, containing information on the way of walking of the subjects. This requires that the network generalizes well to subjects *not seen* at training time, as the performance of the re-identification mechanism strongly depends on the quality of the extracted features. To this end, in this work the NN is trained using a weighted combination of the *cross-entropy loss* [86], denoted by $\mathcal{L}_{\rm ce}$, the *center loss* [89], $\mathcal{L}_{\rm cnt}$, and the *triplet loss* [90], $\mathcal{L}_{\rm tri}$.

The cross-entropy is the most widely used loss for classification purposes in deep learning, and here it is used to train the network to distinguish among the different subjects [86]. However, just training the NN on a classification problem does not lead to sufficiently discriminative features for the re-identification mechanism. The center loss is adopted to additionally force the feature representations belonging to the same class to be close in the feature space, in terms of Euclidean distance. Specifically, denoting by $\boldsymbol{c}_l$ the centroid of the feature vectors belonging to class l, the center loss is

$$\mathcal{L}_{\rm cen}(\boldsymbol{v}, l) = ||\boldsymbol{v} - \boldsymbol{c}_l||_2^2, \tag{5.16}$$

where the centroids are learned as part of the training process via the back-propagation algorithm [89].

The triplet loss is used to push apart the feature representations of inputs belonging to different classes. For this, triplets of input samples are selected from the training set, two of them from the same class, leading to feature vectors $\boldsymbol{v}_a$ and $\boldsymbol{v}_b$, and one belonging to a different class, leading to a third feature vector $\boldsymbol{v}_c$. For further details on the triplet selection process, see Section 3.2 of [90]. The triplet loss is written as

$$\mathcal{L}_{\rm tri}(\boldsymbol{v}_a, \boldsymbol{v}_b, \boldsymbol{v}_c) = \max\left\{||\boldsymbol{v}_a - \boldsymbol{v}_b||_2^2 - ||\boldsymbol{v}_a - \boldsymbol{v}_c||_2^2 + \mu, 0\right\}, \tag{5.17}$$

where μ is a margin hyperparameter, set to 1. Hence, the feature extractor is trained with the following total loss function

$$\mathcal{L} = \mathcal{L}_{\text{ce}} + \mathcal{L}_{\text{tri}} + \omega\mathcal{L}_{\text{cen}}, \tag{5.18}$$

where the parameter $\omega = 0.5$ weighs the relative importance of the center loss. In the implementation, a training dataset containing mmWave RADAR point-clouds from 16 subjects is used. It was collected in different indoor environments to increase the generalization capabilities of the NN. The optimization is carried out using Adam [86] with learning rate 10^{-4} and an L_2 regularization rate of 8×10^{-5} for 250 epochs, as summarized in Tab. 5.1. Hyperparameters tuning was carried out using a greedy search procedure, optimizing the value of the loss $\mathcal{L}$ on a validation set containing a randomly selected subset (20%) of the training data.

Feature extraction

At inference time, i.e., during the system operation, the NN is used to compute feature vectors that are representative of the subjects' gait. Specifically, 45 steps (3 seconds) long sequences of RADAR point-clouds are collected for each tracked subject. The point-cloud sequences are denote by $\boldsymbol{Z}$ in the following. The inner representation $\boldsymbol{v} = \mathcal{F}(\boldsymbol{Z})$, after L_2-normalization, is used as the feature vector for the following re-identification mechanism.

5.4.8 Weighted Extreme Learning Machine (WELM)

The weighted extreme learning machine (WELM) [82] is a particular kind of single-layer feedforward neural network in which the weights of the hidden nodes are chosen randomly, while the parameters of the output layer are computed analytically. Consider an n_{cls}-class classification problem, a training set $\mathcal{V} = \cup_{n=1}^{n_{\text{cls}}} \mathcal{V}_n$ of input *feature vectors* $\boldsymbol{v}$ (see Section 5.4.7), each with an associated one-hot encoded label $\boldsymbol{y} \in \{0, 1\}^{n_{\text{cls}}}$, where $\mathcal{V}_n$ is the set containing the vectors from class $n = 1, \dots, n_{\text{cls}}$. For any $\boldsymbol{v} \in \mathcal{V}$, the WELM computes the matrix of hidden feature vectors $\boldsymbol{H} \in \mathbb{R}^{|\mathcal{V}| \times L}$, with rows $\boldsymbol{h}(\boldsymbol{v})$, where L is the number of WELM hidden units and $\boldsymbol{h}(\cdot)$ is a non-linear activation function. milliTRACE-IR uses $\boldsymbol{h}(\boldsymbol{v}) = \text{ReLU}(\boldsymbol{W}\boldsymbol{v} + \boldsymbol{b})$ where ReLU is the rectified linear unit [86] ($\text{ReLU}(x) = \max(x, 0)$) and $\boldsymbol{W}, \boldsymbol{b}$ are the weights and biases of the ELM hidden layer, respectively. The elements of $\boldsymbol{W}$ and $\boldsymbol{b}$ are here generated from $\mathcal{N}(0, 0.1)$. The WELM learning process amounts to computing, for each class n, the optimal values of an output weight vector $\boldsymbol{\beta}_n$ that minimizes the *weighted* LS L_2-regularized quadratic cost function $||\boldsymbol{H}\boldsymbol{\beta}_n - y_n||_{\boldsymbol{\Omega}}^2 + \lambda||\boldsymbol{\beta}_n||_2^2$, where λ is a regularization parameter and $\boldsymbol{\Omega}$ is a diagonal weighting matrix used to boost the importance of those samples belonging to under-represented classes. This compensates for the tendency of the standard ELM to favor over-represented classes at inference time [82]. In the analyzed scenario, the individuals move freely in the environment across different rooms, so the number of feature vectors collected from each of them is not only unknown in advance, but highly variable. Hence, the training set usually contains unbalanced classes, and milliTRACE-IR uses

$$\Omega_{i,i} = 1/|\mathcal{V}_{n_i}|,\ i = 1, \dots, |\mathcal{V}|, \tag{5.19}$$

where $n_i = \text{argmax}_n(\boldsymbol{y}_i)_n$ denotes the class of the i-th vector. Stacking all the $\boldsymbol{\beta}_n$ into a single matrix $\boldsymbol{B} \in \mathbb{R}^{L \times n_{\text{cls}}}$ and the labels into matrix $\boldsymbol{Y} \in \{0,1\}^{|\mathcal{V}| \times n_{\text{cls}}}$, the WELM output weights $\boldsymbol{B}$ can be computed in closed-form using one of the following equivalent expressions

$$\boldsymbol{B} = \boldsymbol{H}^T \left(\lambda \boldsymbol{I} + \boldsymbol{\Omega} \boldsymbol{H} \boldsymbol{H}^T\right)^{-1} \boldsymbol{\Omega} \boldsymbol{Y}, \text{ or} \tag{5.20}$$

$$\boldsymbol{B} = \left(\lambda \boldsymbol{I} + \boldsymbol{H}^T \boldsymbol{\Omega} \boldsymbol{H}\right)^{-1} \boldsymbol{H}^T \boldsymbol{\Omega} \boldsymbol{Y}. \tag{5.21}$$

Due to the dimension of the matrix to be inverted, if $|\mathcal{V}| > L$, it is more convenient to use Eq. (5.21), while if $|\mathcal{V}| \leq L$ Eq. (5.20) has to be preferred. The output classification for a vector $\boldsymbol{v}$ is then computed as $\text{argmax}_i \left(\boldsymbol{h}(\boldsymbol{v})^T \boldsymbol{B}\right)_i$, where $\boldsymbol{h}(\boldsymbol{v})^T \boldsymbol{B}$ is a vector of WELM scores for each class.

5.4.9 WELM based Person Re-Id

To enable person re-identification based on the feature vectors $\boldsymbol{v}$ extracted by the NN, milliTRACE-IR uses the WELM multiclass classifier of Section 5.4.8, which is *trained at runtime* only when the system has to re-identify a previously seen subject. This is done by sequentially collecting feature vectors from all the subjects seen by the system at operation time, and storing them into the training set $\mathcal{V}$.

Note that, although an online sequential version of the ELM training process has been proposed in [91], the WELM is trained every time a person has to be re-identified using a batch implementation and including in the training set $\mathcal{V}$ all the subjects seen up to the current time-step k. This is because in the online training procedure of [91] the number of classes has to be *fixed* in advance, while in the considered setup the number of subjects seen by the system may change in time and the Re-Id procedure must be flexible to the addition of new individuals to the training set $\mathcal{V}$. The WELM training and re-identification phases are detailed next and in Alg. 5.2.

Training

The training process is performed at runtime as explained in Section 5.4.8, using $L = 1,024$ and $\lambda = 0.1$. During the normal system operation, the feature vectors obtained from each track are continuously added to set $\mathcal{V}$, storing the corresponding one-hot encoded vectors containing the subjects' identities into matrix $\boldsymbol{Y}$. To reduce the computational burden, the feature extraction step is executed every 5 time-steps. This is reasonable, as the input sequences to the NN contain 45 time-steps overall and extracting the features at every time-step would lead to highly correlated, and therefore less informative feature vectors, in addition to entailing a higher computation cost. At time-step k, if a subject has to be re-identified, the training procedure of Section 5.4.8 is executed (lines $1-4$): the WELM feature vectors $\boldsymbol{H}$ are computed by applying the activation function $\boldsymbol{h}(\cdot)$ to each training vector and the weight matrix $\boldsymbol{\Omega}$ is obtained from Eq. (5.19) (lines $1-3$). The WELM output matrix $\boldsymbol{B}$, is computed using Eq. (5.20) or Eq. (5.21) depending on $|\mathcal{V}|$ (line 4).

Algorithm 5.2 Re-identification mechanism at time k.

```
Require: Training set $\mathcal{V}$, track to be re-identified $t^{\mathrm{id}}$.
Ensure: Re-id label of $t^{\mathrm{id}}$.
 1: $\boldsymbol{H} \leftarrow \left[\boldsymbol{h}^T(\boldsymbol{v}), \forall\, \boldsymbol{v} \in \mathcal{V}\right]$
 2: $\boldsymbol{Y} \leftarrow$ labels of $\mathcal{V}$
 3: $\boldsymbol{\Omega} \leftarrow$ Eq. (5.19)
 4: $\boldsymbol{B} \leftarrow$ Eq. (5.21) or Eq. (5.20) depending on $|\mathcal{V}| \lesseqgtr L$
 5: $\boldsymbol{\xi}_0 \leftarrow \mathbf{0}$
 6: for $j = 1, \dots, W$
 7:     $\boldsymbol{v}_j^{\mathrm{id}} \leftarrow \mathcal{F}(\boldsymbol{Z}_j)$
 8:     $\boldsymbol{\xi}_j \leftarrow \left[\boldsymbol{h}^T(\boldsymbol{v}_j^{\mathrm{id}})\boldsymbol{B} + j\boldsymbol{\xi}_{j-1}\right]/(j+1)$
 9: end for
10: label $\leftarrow \arg\max_i (\boldsymbol{\xi}_W)_i$
```

Re-identification

The Re-Id procedure is used to recognize subjects that have been seen by the system and associate them with their temperature measurement and their past movement history in the monitored area. Denoting by t^{id} the track to be re-identified, the trained WELM processes the NN features of this user, $\boldsymbol{v}^{\mathrm{id}}$, as follows: $\boldsymbol{h}(\boldsymbol{v}^{\mathrm{id}})^T\boldsymbol{B}$. Due to the high variability of human movement, rather than considering a single feature vector, milliTRACE-IR computes the cumulative average WELM scores over a time window of length W, where the average score at time $j = 1, \dots, W$ is referred to as $\boldsymbol{\xi}_j$ (lines $6-9$). The identity label corresponds to the index of the largest element of $\boldsymbol{\xi}_W$ (line 10).

5.5 Experimental Results

In this section, the experimental results obtained by testing the system in different indoor environments are presented.

5.5.1 Implementation

Hardware. milliTRACE-IR has been implemented on an NVIDIA Jetson TX2 edge computing device[‡], with 8 GB of RAM and a NVIDIA Pascal GPU. The Jetson TX2 has been connected via USB to a Texas Instruments IWR1843BOOST mmWave RADAR[§], operating in the $77-81$ GHz band, and via Ethernet to a FLIR A65 thermal camera[¶], as shown in Fig. 5.7. The experiments have been performed in real-time at a frame rate of $1/\Delta = 15$ Hz.
The RADAR device operates in FMCW mode, using a chirp bandwidth $B = 3.07$ GHz, which leads to a range resolution of $c/2B = 4.88$ cm, and 64 chirps per sequence, obtaining a maximum measurable velocity of 4.77 m/s and velocity resolution of 14.92 cm/s.
The thermal camera has a 640×512 focal plane array (FPA), a spectral range of $[7.5, 13]$ μm, a

[‡]https://developer.nvidia.com/embedded/jetson-tx2
[§]https://www.ti.com/tool/IWR1843BOOST
[¶]https://www.flir.it/products/a65/

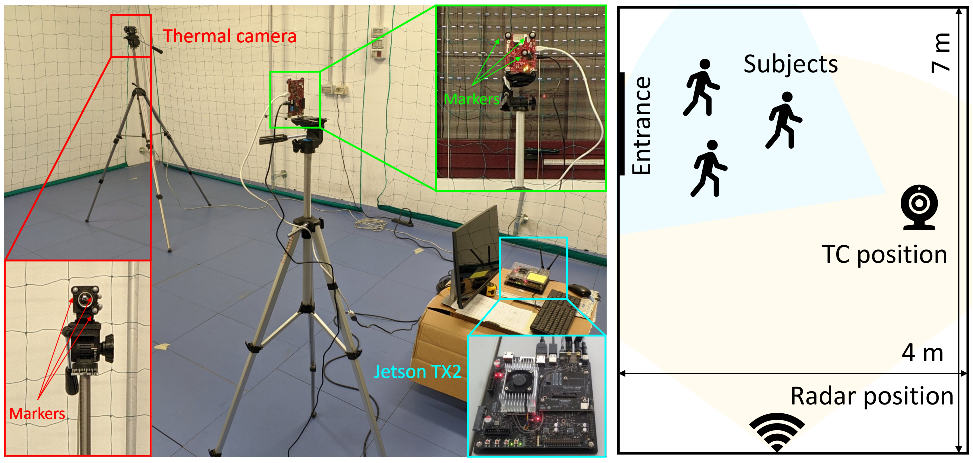


Figure 5.7: Experimental setup for the data association.

temperature range of $[-25, 135]°C$, a measurement uncertainty of $\pm 5°C$, and a noise equivalent temperature difference (NETD) of 50 mK.

Software. The system has been developed in Python, using the NumPy, SciPy and OpenCV libraries for the implementation of the tracking phases (for RADAR and thermal camera) and the proposed data association (Section 5.4.5), clustering (Section 5.4.4) and re-identification (Section 5.4.9) algorithms. Tensorflow and Keras libraries have been used to implement the feature extraction NN (Section 5.4.7). The pre-trained face detector for the thermal images (Section 5.4.2) has been taken from the open-source YOLOFace[‖] implementation.

5.5.2 TC and Radar Tracks Association

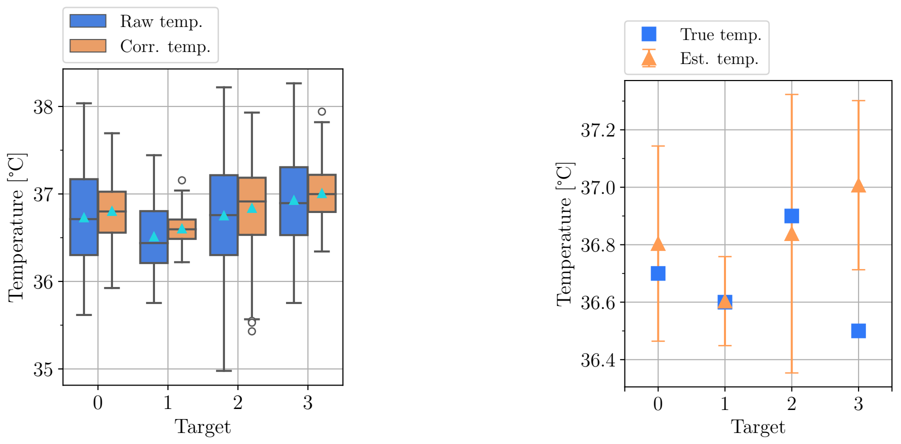
To assess the performance of the RADAR-TC track association method, experimental tests were conducted in a 7×4 m research laboratory. A motion tracking system including 10 cameras was used to gather ground-truth (GT) data about the locations of the subjects, by placing markers atop their heads. This camera based tracking system provides 3D localization with millimiter-level precision, for all markers, at a rate of 100 Hz. The RADAR and the TC were placed as shown in Fig. 5.7. 5 measurement sequences with 2 subjects and 9 sequences with 3 subjects, all freely entering the room, were collected. The roto-translation matrix $\mathbf{\Phi}$ was estimated using a set of markers applied to the devices, while the TC intrinsic matrix $\mathbf{\Psi}$ (see Section 5.4.5) and the radial distortion coefficients were obtained through the Zhang's method [92], using a sun-heated checkerboard pattern.

An *association* is defined as a specific pairing $i \leftrightarrow j$ of a track i from the RADAR with a track j from the TC, and a *correct association* as an association for which the two tracks correspond to the same subject. Given a set of tracks, the set of all the correct associations performed by the algorithm is denoted by $\mathcal{A}_{\mathrm{TP}}$ (*true positives*), the set of all the associations performed by the algorithm as $\mathcal{A}_{\mathrm{P}}$ (*positives*), and the set of all the associations that the algorithm should have

[‖]https://github.com/sthanhng/yoloface

	With $\rho(K)$		Without $\rho(K)$	
	Pr [%]	Rec [%]	Pr [%]	Rec [%]
$A_x + A_d$	**97.3**	**97.3**	91.9	91.9
A_x only	91.9	89.2	89.7	94.6
A_d only	92.1	94.6	86.8	89.2

Table 5.2: Impact of the components of the cost function. Row labels A_x, A_d, and $A_x + A_d$ indicate, respectively, that only costs A_x, A_d or the sum of the two were used in the evaluation. Label "With $\rho(K)$" indicate that the corrective term, $\rho(K)$, was used, while label "Without $\rho(K)$" means $\rho(K) = 1$.



(a) Comparison between with (*Corr. temp.*) and without (*Raw temp.*) distance-based correction. The triangles show the mean values.

(b) Comparison between the estimated temperatures and the true temperatures. The error bars represent the standard deviations.

Figure 5.8: Results of the temperature screening.

performed, based on the GT, as $\mathcal{A}_{\mathrm{R}}$ (*relevant*).

To quantify the association performance of the system, define the *precision*, $\mathrm{Pr} = |\mathcal{A}_{\mathrm{TP}}|/|\mathcal{A}_{\mathrm{P}}|$, and the *recall*, $\mathrm{Rec} = |\mathcal{A}_{\mathrm{TP}}|/|\mathcal{A}_{\mathrm{R}}|$. Using these metrics, the proposed track association method is evaluated by assessing the contribution of each cost component in $A(i, j)$ (see Eq. (5.11)). The results are reported in Tab. 5.2, where the row labels A_x, A_d, and $A_x + A_d$ indicate the cost function used. The table also shows the impact of adding the correction coefficient $\rho(K)$ (see Eq. (5.10)): for the case "Without $\rho(K)$", $\rho(K)$ is set to 1.

As shown, the proposed track association method reliably associates the RADAR and TC tracks, reaching precision and recall both higher than 97%. The joint use of A_x, A_d and $\rho(K)$ leads to improvements of up to 11% and 8% for the precision and recall metrics, respectively.

5.5.3 Temperature Screening

Remarkably, the proposed temperature screening method does not require people to stand in front of the TC sensor, but estimates their temperature as they move within the FoV of the TC. In order for the method to return accurate temperature measurements, the subject' frontal face should be

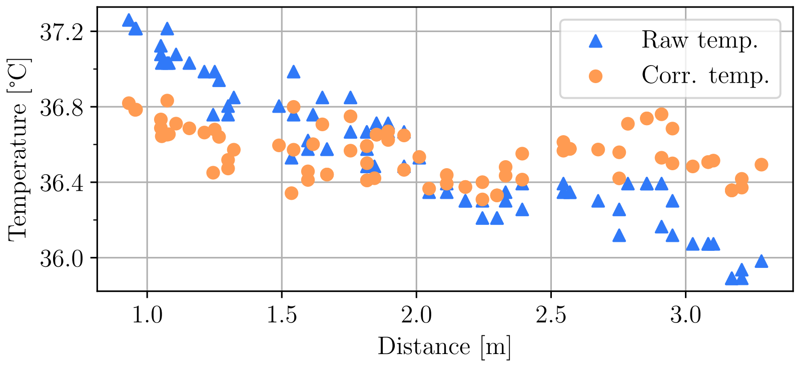


Figure 5.9: Temperature measurements from a subject moving in front of the TC with (*Corr. temp.*) and without (*Raw temp.*) distance-based correction.

	Mean [°C]	± std [°C]	True temp. [°C]	Error [°C]
Target 0	36.8	0.340	36.7	0.104
Target 1	36.6	0.155	36.6	0.004
Target 2	36.8	**0.485**	36.9	−0.062
Target 3	37.0	0.294	36.5	**0.507**

Table 5.3: Results of the temperature estimation and comparison with respect to the true values for the 4 targets. The worst cases are highlighted.

captured by the TC for a minimum time duration. For this reason, it is advisable to place the TC near a point of passage, e.g., in proximity of an entrance. The temperature screening method was tested on $4-7$ sequences of ~ 10 s each were collected from 4 different individuals moving within 3.5 m from the TC. Each subject was tested at a different time of the day, to gauge the effects of the changing (thermal) environmental conditions, and of a possible concept drift (e.g., heating) of the TC after a long period of operation. Furthermore, as explained in Section 5.4.3, a linear function $\alpha(\cdot)$ was fit to compensate for the influence of the distance on the measures.

To evaluate the benefit brought by the correction based on the targets' distance, in Fig. 5.8a, the results obtained with (*Corr. temp.*) and without (*Raw temp.*) the correction are compared. Since the TC is intrinsically subject to a bias, to facilitate the comparison of the measures, in the *Raw temp.* case only this bias is corrected, assuming a constant target distance of 2 m and multiplying each measured temperature by $\alpha(2) = a_0 + 2a_1$. The full method (*Corr. temp.*), instead, uses the rescaled average estimate, as per Eq. (5.15). The box-plot shows that the range of the corrected temperatures is significantly reduced (for these experiments, the true temperature is constant), demonstrating the efficacy of the proposed correction plus averaging approach. As an illustrative example, Fig. 5.9 shows the impact of the distance-based correction on data measurements from a subject moving in front of the TC.

Fig. 5.8b compares the temperature estimates from milliTRACE-IR and the true temperatures measured with a contact thermometer. The numerical results are reported in Tab. 5.3, where

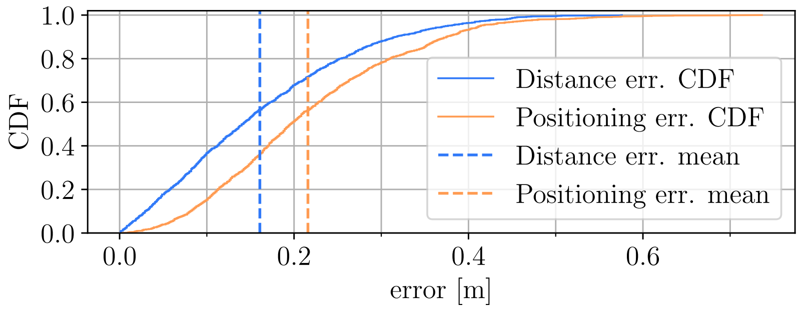


Figure 5.10: CDF of the absolute error between the true (ground truth) and the estimated subject's position / inter-subject's distance, as measured by the RADAR tracking system. The dashed lines denote the mean error.

the worst cases are reported in bold fonts. Mean temperatures are estimated with a maximum standard deviation from the mean smaller than 0.5 °C and a maximum absolute error with respect to the true temperature of about 0.5 °C. Note that only one of the subjects in Fig. 5.8b exhibits this maximum error (subject 3), while the absolute error for the others remains within 0.1 °C. These errors descend from the fact that the environmental conditions and the heating of the thermal camera affect the measurements in an unpredictable way, modifying the *bias* of the fitting function. Notwithstanding, the thermal screening capability of milliTRACE-IR is significantly better than that of existing approaches, see Section 5.5.7. Also, some improvements are possible by, e.g., applying a correction based on an external reference, such as a piece of material instrumented with a contact thermometer and located within the field of view of the TC, or monitoring the statistics of the people's temperature (mean μ and standard deviation σ) to detect anomalous samples within such empirical distribution. For instance, an alarm could be raised for those subjects whose temperature is greater than $\mu + c \times \sigma$, for a user-defined threshold c. This would allow the system to continuously and autonomously adapt to different operating conditions.

5.5.4 Positioning and Social Distance Monitoring

To evaluate the performance of the RADAR tracking system in estimating the position of the targets and the inter-subject's distance, tests were conducted in the 7×4 m research laboratory described in Section 5.5.2. A total of 7 sequences of duration $10-15$ s were collected, each with 3 subjects moving freely in the room, along with their GT locations obtained from the motion tracking system. The root mean squared error (RMSE) between the mmWave RADAR estimated locations and the GT is used as a performance metric. Moreover, the inter-subject distances were measured, considering all the possible combinations of the three subjects and leading to a total of 21 inter-subject distances across all the recorded sequences.

The cumulative distribution functions (CDF) of the absolute error between the ground truth and the estimated subject's position/inter-subject distance, as measured by the RADAR tracking system, is shown in Fig. 5.10, along with the corresponding mean values. The numerical results are provided in Tab. 5.4. The RADAR system achieves an absolute *positioning* error within 0.3 m in 80% of the cases. For the inter-subject *distance*, the error remains within 0.25 m in 80% of the cases.

	Mean [m]	± std [m]	Frames	Time [s]
Position RMSE	0.216	0.115	1448	97
Subj. distance RMSE	0.161	0.112	1153	77

Table 5.4: RMSE of the subject's position and of the inter-subject's distance estimated by the RADAR sensor, computed against the GT.

	milliTRACE-IR		**DBSCAN**	
	r_{cl} [%]	corr. tracked	r_{cl} [%]	corr. tracked
2 sub. parallel	90.7	✓	46.5	×
2 sub. crossing	87.9	✓	59.6	×
2 sub. close	89.9	✓	69.7	✓
3 sub. parallel	92.3	✓	65.3	✓
3 sub. crossing	83.7	✓	73.5	×

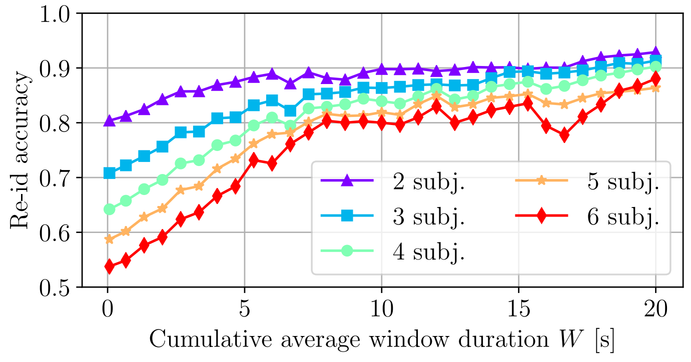
Table 5.5: Ratio r_{cl} between the number of frames in which the different subjects are correctly separated and the total number of frames, using the proposed method and DBSCAN. Symbols "✓" and "×" denote success and failure of the tracking step, respectively.

5.5.5 Effectiveness of the Improved Clustering Technique

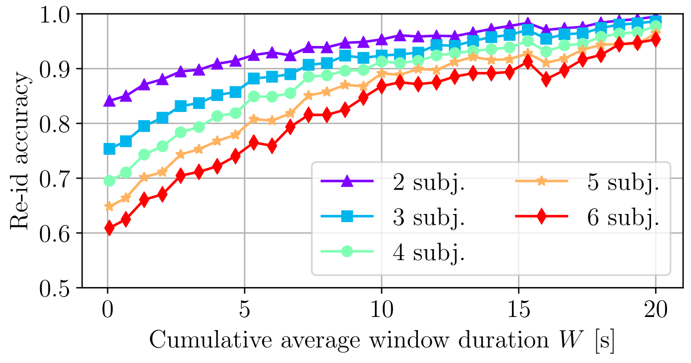
To evaluate the improvement brought by the proposed clustering method over the standard DBSCAN, both algorithms were tested on specific measurement sequences with subjects moving within 1 m from one another. To quantify the clustering performance, the correct clustering ratio, r_{cl}, is used. This metric represents the fraction of frames in which the clusters belonging to the different subjects are correctly separated. The results of this evaluation are summarized in Tab. 5.5. The evaluation is conducted on sequences with 2 and 3 individuals *(i)* walking along parallel paths with the same velocity and at a distance between 0.5 m and 0.8 m (*parallel*), *(ii)* walking along crossing paths, with subjects coming as close as 0.2 m from one another (*crossing*) and *(iii)* staying still and moving arms at an inter-subject distance of approximately 0.8 m (*close*). The proposed clustering algorithm led to a large improvement (up to 44 %) in terms of r_{cl} metric with respect to DBSCAN. In addition, for 3 of the 5 test sequences, DBSCAN led to failures in the tracking process, either merging the tracks of different subjects, or failing to detect some of them, while milliTRACE-IR correctly tracked all the subjects in all cases.

5.5.6 Person re-identification

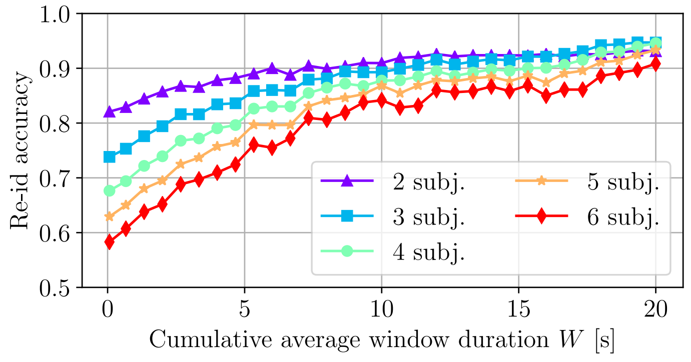
The proposed WELM based Re-Id algorithm was evaluated on a set of mmWave RADAR measurements from 6 individuals who were *not* included among the 16 subjects used to train the feature extraction NN. The tests were conducted in a 12×3 m research lab, with furniture that made the evaluation challenging. The training data contains 4 minutes of measurements ($3,600$ RADAR frames) while over 1 minute of measurements per subject ($1,000$ frames) was used as test data. In both the training and the test data, the individuals walked freely in the room. The RADAR



(a) 1 min. training data.



(b) 3 min. training data.



(c) Imbalanced training data.

Figure 5.11: Re-identification accuracy results. In (a) and (b) the re-identification algorithm is used with 1 and 3 minutes of training data per subject, respectively. In (c), 1 minute of training data was used for a randomly selected subset containing half of the subjects, while 4 minutes were used for the remaining half.

position was changed for each test to gauge the impact of varying the RADAR point-of-view.

Re-Id accuracy. The Re-Id accuracy as a function of W (see Alg. 5.2) is shown in Fig. 5.11a and Fig. 5.11b. The curves of these plots are obtained averaging the results of 20 different WELM initializations, and all the possible combinations of the considered number of subjects (from 2 to 6) over the 6 total individuals. As expected, the Re-Id performance increases with an increasing inference time (larger W) and with the length of the training sequences: the accuracy gain is

	WELM			**CS baseline**		
	1 min.	4 min.	imb.	1 min.	4 min.	imb.
$W = 0$ s	53.8	60.9	58.3	44.6	49.5	51.6
$W = 10$ s	80.0	86.8	84.2	63.9	77.7	80.6
$W = 20$ s	88.6	95.3	90.8	72.2	88.8	86.9

Table 5.6: Re-Id accuracies obtained by the WELM and the CS baseline on 6 subjects using 1 and 4 minutes balanced training sets, and an imbalanced training set. The cumulative average window W is set to 0 s (a single test feature vector is used), 10 s or 20 s.

about 10% by going from 1-minute (Fig. 5.11a) to 3 minutes (Fig. 5.11b) long training sequences. Also, milliTRACE-IR reaches high Re-Id accuracy using $W \geq 15$ s and the detrimental effect of increasing number of subjects to be classified is greatly reduced using larger values of W, as accumulating the WELM scores over longer time windows increases the robustness of the WELM decision. Overall, the accuracy of the proposed method is higher than 95% in all cases, only using 3 minutes of training data per subject and $W = 20$ s, which are reasonable in practice. The worst-case (3 minutes of training data for 6 subjects) WELM training time, on the ARM Cortex-A57 processor of the Jetson TX2 device, took 2.98 ± 0.015 s.

Impact of imbalanced training data. As shown in Fig. 5.11c, the effect of imbalanced training data is successfully mitigated by the sample weighting strategy of Eq. (5.19). In this evaluation, the WELM was trained with 1 minute of data for a randomly selected subset containing half of the subjects and 4 minutes for the remaining half.

Improvement over a baseline. Tab. 5.6 compares the WELM to a baseline classification method widely used in camera-based person Re-Id [76] that, unlike milliTRACE-IR, does not learn a similarity score based on the actual distribution of the feature vectors at operation time. The baseline algorithm collects the training feature vectors along with the corresponding labels and computes the *centroid* of each class m in the NN feature space, denoted by $\boldsymbol{c}_m$. To re-identify a subject, the *cosine similarity* (CS) between his/her feature vectors, $\boldsymbol{v}$, and the centroid of each class m is computed, obtaining a similarity score $s_m = \boldsymbol{c}_m^T \boldsymbol{v}/(||\boldsymbol{c}_m||_2 \times ||\boldsymbol{v}||_2)$, and the classification is performed taking $\text{argmax}_m\, s_m$. The WELM outperforms the baseline scheme in all the tests, see Tab. 5.6. The performance gap is significant for little training data (up to 16% improvement), small windows and imbalanced training sets.

5.5.7 Comparison with existing approaches

In this section a comparison between milliTRACE-IR and available methods from the literature is provided. To the best of the authors' knowledge, only two works exploit both mmWave RADARs and thermal cameras to perform human sensing and/or temperature screening, namely, the works from Ülrich et al. [7] and Savazzi et al. [63]. Since none of the two tackles all the points that milliTRACE-IR addresses, they are here considered, separately, to compare different aspects. The data association strategy is compared with that proposed in [7], while [63] is used to compare the positioning, distance monitoring, and temperature screening parts. In Tab. 5.7, symbols "×" and

	milliTRACE-IR	**Ülrich** [7]	**Savazzi** [63]
Positioning range RMSE [m]	0.19	×	0.45
Interpersonal dist. RMSE [m]	0.17	×	0.5*
Positioning angle RMSE [°]	3.1	×	7.0
Thermal screening RMSE [°C]	0.13	×	0.45
Thermal screening range [m]	3.5	×	1.1
Per-frame assoc. Pr [%]	98.6	25.2	n.a.
Per-frame assoc. Rec [%]	98.5	78.4	n.a.
Re-identification acc. [%]	≈ 90	×	×

Table 5.7: Comparison of milliTRACE-IR with the works from Ülrich et al. [7] and Savazzi et al. [63]. Symbols "×" and "n.a." denote, respectively, that the task is not tackled or that there is no available result for the considered quantity in the original papers. The symbol "*" is used to highlight that the value is not an RMSE value but the minimum interpersonal distance threshold considered in [63].

"n.a." denote, respectively, that the task is not tackled or that no specific result is provided in the corresponding work.

Data association. In [7] (Ülrich et al.), people are detected in thermal images by applying the Viola and Jones algorithm [93] to detect the upper bodies of the subjects in the environment. The distance between the TC and each subject is roughly retrieved from the dimension of the bounding box enclosing the upper body of each person, similarly to what milliTRACE-IR does with faces. The TC detections are then associated, on a frame basis, with range measurements obtained with a mmWave RADAR by minimizing a Gaussian-shaped association cost. This cost provides an estimate of the probability that the corresponding association is correct, based on the difference between the distance estimates by the TC and by the mmWave RADAR. This data association method has been implemented and tested on the dataset of Section 5.5.2, comparing it to the data association strategy of milliTRACE-IR. For a fair comparison, the YOLOv3 detector has been used in place of the Viola and Jones algorithm, as, besides providing superior performance, it is the same detector used by milliTRACE-IR. This guarantees that any difference in the data association results is only due to the data association strategy. At every time frame, each bounding box has been associated with the RADAR detection yielding the highest association probability, which corresponds to the smallest difference in the two distance estimates. The main differences between the approach in [7] and that of milliTRACE-IR are that, in [7]: *(i)* the association is per-frame and not per-track, *(ii)* the estimated distance from the TC is the only feature considered for the association, and *(iii)* the Hungarian algorithm is not used, so different bounding boxes can be, erroneously, associated with the same RADAR detection. Numerical results for the precision ("Pr") and recall ("Rec") metrics are presented in Tab. 5.7. Since the association technique of [7] performs a per-frame association, the table shows the per-frame performance of milliTRACE-IR, computed by counting the number of frames that are correctly classified using milliTRACE-IR's per-track association algorithm. From these results, it can be seen that milliTRACE-IR performs notably better in associating mmWave RADAR with TC human detections. The largest improvement is brought by the combination of milliTRACE-IR *per-track* association paradigm with the Hungarian

algorithm, which effectively filters out ghost tracks and spurious detections which often occur in real world scenarios, significantly boosting the robustness of the scheme.
Positioning, distancing, and temperature screening. In [63] (Savazzi et al.), people localization, interpersonal distance monitoring, and temperature screening are addressed using thermopiles and mmWave RADARs. Since in [63] the data association strategy is not disclosed, a comparison is here provided only for the previously mentioned tasks. In the chapter, positioning performance is evaluated in terms of range (radial distance) and angular RMSEs. Numerical values for these metrics are given in Tab. 5.7 considering the dataset of Section 5.5.4 for milliTRACE-IR and the (average) values from Tab. II of [63] for their algorithm.

In the same work, interpersonal distance monitoring is obtained by dividing the monitored area into a regular grid, whose cells have a side length of 0.5 m. The system is able to distinguish subjects occupying adjacent cells, which are considered to be violating the minimum interpersonal distance of 1 m, thus raising an alarm. For this reason, the resolution of the method of [63] is 0.5 m in the best case (a lower bound for the interpersonal distance estimation error). In Tab. 5.7, this value is reported alongside the RMSE of milliTRACE-IR in measuring interpersonal distances, marking the former with a "*" symbol, to highlight that it is not an RMSE.
Thermal screening performance comparisons are also presented in Tab. 5.7, where "Thermal screening range [m]" refers to the maximum distance at which the tests were carried out. milliTRACE-IR performs better than [63] in all the considered tasks, showing a larger monitoring range and more accurate body temperature estimates. In addition, milliTRACE-IR combines these monitoring capabilities with a robust data association strategy and with the capability to re-identify subjects when moving through different areas.

5.6 Concluding Remarks

This chapter presented the design and implementation of milliTRACE-IR, the first system combining high resolution mmWave RADAR devices and infrared cameras to perform non-invasive joint temperature screening and contact tracing in indoor spaces. This system uses thermal cameras to infer the temperature of the subjects, achieving measurement errors within 0.5 °C, and mmWave RADARs to infer their spatial coordinates, by successfully locating and tracking subjects that are as close as 0.2 m apart. This is possible thanks to improvements along several lines, such as the association of the thermal camera and RADAR tracks from the same subject, along with a novel clustering algorithm combining density-based and Gaussian mixture methods to separate the RADAR reflections coming from different subjects as they move close to one another. Moreover, milliTRACE-IR performs contact tracing: a person with high body temperature is reliably detected by the thermal camera sensor and subsequently traced across a large indoor area in a non-invasive way by the RADARs. When entering a new room, this subject is re-identified among several other individuals with high accuracy (95%), by computing gait-related features from the RADAR reflections through a deep neural network and using a weighted extreme learning machine as the final re-identification tool.

6
An Experimental Prototype for Multistatic Asynchronous ISAC

In this chapter, we move a first step towards the use of mmWave RADAR networks in an ISAC setting. The idea is to study how person's movement is perceived differently based on their relative position with respect to TX/RX devices, with the aim of exploiting spatial diversity in data fusion.

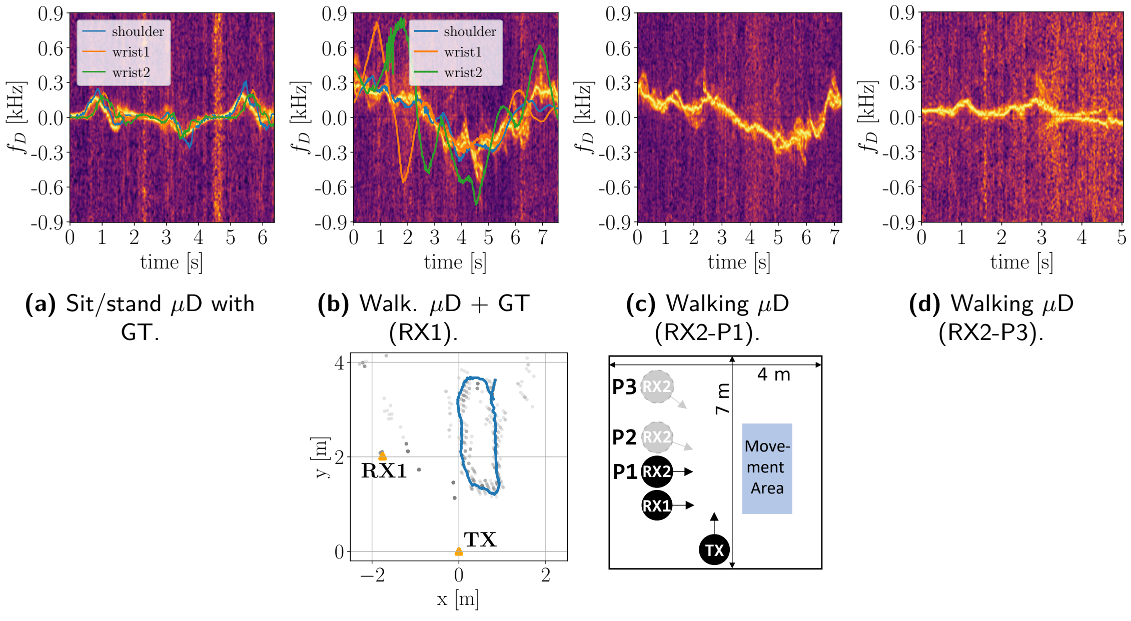
6.1 Introduction and related work

ISAC reuses wireless network devices for sensing the surroundings by applying the radar principles to communication signals. The strength of this approach lies in the ubiquity of wireless communication devices which avoids the costly deployment of dedicated sensors and allows exploiting spatial diversity and boost sensing accuracy thanks to *multistatic* settings [94]. However, communication devices are based on separated TX and RX nodes with *asynchronous* local oscillators (LOs), which make conventional radar techniques unusable [95]. Several methods exist to mitigate the impact of such asynchrony [95], but none have been experimentally validated on multistatic scenarios.

We fill this gap by presenting the first experimental prototype of *asynchronous and multistatic* ISAC in the mmWave frequency band, based on the IEEE 802.11ay standard at 60 GHz. The LO asynchrony is compensated for by exploiting the LoS propagation of the signal between each TX and RX pair. Our results on human movement sensing show that the system can accurately track people and estimate their micro-Doppler (μD) signature from multiple points of view, providing enriched and more reliable movement features for advanced sensing applications (e.g., activity recognition). Finally, our measurements are supplemented with marker-based motion tracking GT data, providing, for the first time, a reliable reference to assess both μD and tracking traces.

6.2 System Overview

The system is composed of one TX and two RX antenna arrays exchanging standard-compliant single-carrier IEEE 802.11ay data packets. Every packet includes 12 beam-training fields, each transmitted using a different antenna beam pattern (BP), b, that are used to estimate the channel impulse response (CIR). The CIR estimates are processed to remove the TO and the CFO caused by the clocks asynchrony and are used to track the movement of a subject and to extract its μD signature.



(a) Sit/stand μD with GT.

(b) Walk. μD + GT (RX1).

(c) Walking μD (RX2-P1).

(d) Walking μD (RX2-P3).

(e) Example of tracking.

(f) Setup scheme.

Figure 6.1: Experimental results and measurement setup scheme.

6.2.1 Channel Model

At a specific time k, the estimated CIR is represented as a vector of L complex channel gains, called *taps*, and indicized by ℓ. Each tap corresponds to a specific propagation delay (and, therefore, path length) of the transmitted signal. Indicating with T the time between two consecutive channel estimations, with $N_\ell(kT)$ the number of reflections corresponding to tap ℓ, and with $A_n(kT)$ the complex signal attenuation coefficient, the CIR is expressed as

$$h(k,\ell) = \sum_{n=1}^{N_\ell(kT)} A_n(kT) e^{j2\pi[f_{D,n}(kT)+f_{\text{off}}(kT)]kT}, \tag{6.1}$$

where $f_{D,n}(\cdot)$ and $f_{\text{off}}(\cdot)$ represent, respectively, the Doppler frequency induced by targets motion in path n and the CFO. Since training fields are transmitted with different BPs, corresponding to different directions, each of them yields a different CIR, denoted by $h_b(k, \ell)$.

6.2.2 Target tracking and μD extraction

TO compensation. As explained in [96], it is fundamental to compensate for the TO in order to correctly localize a target. In this work, we assume that a LoS path between TX and RX is available and we use it as a common reference to align the CIR estimates in time. For that, according to a dynamic threshold, we detect the peaks in the CIR magnitude and select the first one, as the LoS always corresponds to the shortest path. After that, the CIR is shifted to make the LoS peak aligned with the first CIR tap. This operation is repeated for all time steps k and all BPs b.
Target detection. After this operation, to facilitate the detection of dynamic targets, static paths are removed from the CIRs. Since static paths are constant across time, we consider the time-averaged CIR, denoted as $\overline{h}_b(\ell)$, to be a good estimate of the static background. Then, the foreground CIR amplitude is computed as $\tilde{h}_b(k, \ell) = \max(|h_b(k, l)| - \overline{h}_b(l), 0)$. At this point, dynamic target detections at time k are computed by applying a peak detection algorithm to $\sum_{b=0}^{N_b-1}(\tilde{h}_b(k, \ell))^2$, being N_b the number of BPs. Finally, assuming the locations of TX and RXs are known, exploiting bistatic geometry [96] we compute the distance of the target from TX and the path's angle of departure, thus, localizing the target.
Target tracking. Based on the detected target locations, an EFK is built to track the movement of the targets, assuming they approximately follow a constant velocity model.
CFO compensation and μD extraction. To compute the μD spectrograms, we utilize multiple subsequent received packets. However, since the TX and RXs LOs are not precisely synchronized, every received packet is affected by a different CFO that spoils the μD reconstruction. Noting that every tap of a CIR have the same CFO, it is possible to isolate the CFO from a static path and use it to correct all paths. In our case, we use the LoS path as a reference to extract the CFO. Let $\phi_{\text{off}}(k) = 2\pi f_{\text{off}}(kT)kT$ be the phase of the LoS path at time k. Then, the CFO-corrected CIR is given by $h'_b(k, \ell) = e^{-j\phi_{\text{off}}(k)} h_b(k, \ell)$. After the correction, the resulting CIR can be used to compute the target's μD spectrogram as in [96].

6.3 Experimental Results

Implementation and Dataset. The experimental setup is a customization of MIMORPH [97], based on the 60 GHz IEEE 802.11ay standard. It comprises 2 Xilinx Zynq UltraScale+ RFSoC ZCU111 boards (one for TX and one for RXs) and 3 Sivers EVK06002/00 antenna arrays with a radio frequency bandwidth of 1.76 GHz. TX and RXs are disposed according to 3 possible different setups, as shown in Fig. 6.1f. TX and RX1 are always positioned at the same locations, while RX2 is located at P1, P2, or P3, based on the setup. RXs point towards the movement area, to maximize the received signal-to-noise ratio of target's reflections. We acquired data from both RXs

simultaneously. In total, we collected 72 sequences, 24 per setup, 36 per array. Within each setup, the subject performed two activities: *walking* along an oval trajectory or *sitting down/standing up* from a chair. Moreover, we acquired *ground-truth* data through a marker-based motion capture system with millimeter level accuracy. Markers were placed on the antenna arrays and on some subject's body parts, namely: wrists, head, and shoulder. We use the head markers to precisely identify the location of the subject, the shoulder markers as an estimate of the chest movement, and the wrist markers as indicative of the arms movement.
Results. Fig. 6.1b-(d) show the μD obtained from the same walking activity when reflections are captured at different locations. We observe that the frequency of the Doppler shift decreases in P2 and P3. Since the bistatic angle β (the angle between TX and RX with vertex at the target) affects the Doppler frequency through a multiplicative factor $\xi = \cos(\beta/2)$, moving RX2 to P2 and P3 corresponds to approaching $\beta = \pi$, and, therefore, ξ decreases. Hence, our observation is in line with what expected from the theory. Fig. 6.1a-(b) show the μD reconstruction along with the estimate from the GT data. In Fig. 6.1a, with sit/stand activity, body parts undergo a similar movement and this is reflected in both μD and GT curves having the same shape. In Fig. 6.1b, the strongest μD component follows the movement of the chest, the largest body part, while the wrists movement align with the weaker components. Fig. 6.1e shows a sample tracking result, where in blue we show the EKF output.

6.4 Conclusions

In this work, we presented the first experimental setup for *asynchronous and multistatic* ISAC in the mmWave frequency band. We collected a dataset featuring 1 TX and 2 RXs simultaneously receiving IEEE 802.11ay data packets. The dataset is supplemented with GT data that allows comparing, for the first time, μD traces with a quantitative and reliable reference. We show that LoS-based asynchrony resolution allows for accurate multistatic sensing without any kind of synchronization. This paves the way to new research to combine the information received at different nodes.

7 Concluding remarks

Contactless sensing of human movements using RF signals has the potential of becoming a game-changer in the way we interact with technology in our daily life, enabling uncountable applications ranging from elderly care, to indoors monitoring, and to autonomous driving.

Up to now, researchers mostly adopted a single-sensor approach to develop sensing with mmWave RADARs. However, we argue that only the combination of *multiple* sensors can provide the sensing capabilities to face real-life scenarios.

In this thesis, we contributed to advance in the field of human sensing with mmWave systems developing tools for the practical deployment of RADAR networks in real-word conditions and for the cooperation of mmWave RADAR sensors among themselves and with other sensors.

In the first part, we addressed the problem of *self-calibrating* a RADAR network and developing methods to integrate multiple RADARs and manage them. The self-calibration approach, presented in Chapter 2, leverages the trajectories of people moving freely in the monitored area to automatically estimate the RADARs relative position and orientation, with no human intervention. Then, in Chapter 3, we developed an experimental testbed for the easy deployment and testing of mmWave RADAR networks.

In the second part, we focused on algorithms for data fusion in mmWave monitoring systems. Exploiting the work from Chapters 2 and 3, in Chapter 4 we developed a system that enables unified and seamless tracking across the nodes of a RADAR network in a semi-distributed fashion. Each radar performs tracking independently of the others and then share with a central unit only its lightweight tracks information, with low network overhead. The central unit, by leveraging the position estimates from the method in Chapter 2, fuses the RADARs' tracks and provides the unified tracking, in real time. Chapter 5 concerned the development of a crowd monitoring system involving the use of a mmWave RADAR and a TC. The monitoring consisted in *(i)* tracking people from radar measurements with the possibility of *(ii)* re-identifying subjects across different indoor spaces through a gait features-based deep learning approach, and, concurrently, *(iii)* associating temperature readings to people tracks. The work also featured a data *fusion* approach to correct thermal estimates through the RADAR-measured subjects distance. Finally, in Chapter 6, we presented some preliminary results regarding RADAR networks in an ISAC setting. The idea is to study how person's movement is perceived differently based on their relative position with respect to the devices, with the aim of exploiting spatial diversity in data fusion.

7.1 Future Research Directions

We identify the following research directions.

Autonomous Configuration of RADAR networks. The employment of RADAR networks effectively helps mitigating for the occlusion problem in crowded spaces and provides enriched sensing capabilities thanks to spacial diversity. In Chapter 2, we presented a self-calibration method that allows to automatically estimate the relative position and orientation of RADARs in a RADAR network. However, this procedure requires some conditions to be satisfied. Specifically, it needs to know in advance which are the nodes to be calibrated and which RADAR to take as a common reference system before computing the transformations. While these information may be straightforward in small and simple scenarios, like a single small room, they are not in more general settings. For example, if the monitoring RADAR network spans an entire building involving hundreds of RADAR sensors, only those illuminating the same area need to calibrated together. For these reasons, automatically configuring a RADAR network demands for methods to *(i)* automatically cluster network nodes into collaborative groups, *(ii)* organize them hierarchically, and, within each cluster, *(iii)* estimate their relative positions and orientations. A solution to point *(iii)* is proposed in Chapter 2, whereas points *(i)* and *(ii)* remain unsolved.

RADAR Networks in the context of ISAC. ISAC appears as a possible way to achieve ubiquitous sensing without the need of deploying new devices, by exploiting hardware already present for communication purposes. Empowering communication devices with RADAR-like sensing capabilities would lead, in fact, to an extremely large RADAR network. However, such network would have some important intrinsic differences with respect to standard RADAR networks. Firstly, mmWave RADAR networks are usually composed of monostatic RADARs, while, in ISAC scenarios, *bistatic* or *multistatic* configurations are the most common. This poses several challenges in terms of synchronization, as LOs of communication devices are usually *asynchronized*, introducing TO and CFO that prevent sensing applications with standard methods. Chapter 6 partially tackles this problem, but its applications are limited, as it requires the constant presence of a LoS path between TX and RX. Secondly, building on top of a communication network constituted of access points and user equipments, nodes of an ISAC RADAR network are intrinsically *mobile*. Nodes mobility represent an important challenge as, from a RADAR perspective, it spoils almost any sensing parameter across time. Therefore, algorithms for its compensation need to be developed.

References

[1] B. P. A. Rohman, M. T. Rudrappa, M. Shargorodskyy, R. Herschel, and M. Nishimoto, “Moving human respiration sign detection using mm-wave radar via motion path reconstruction,” in *2021 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*, 2021, pp. 196–200.

[2] G. Paterniani, D. Sgreccia, A. Davoli, *et al.*, “Radar-based Monitoring of Vital Signs: A Tutorial Overview,” Mar. 2022.

[3] J. Pegoraro and M. Rossi, “Real-time people tracking and identification from sparse mm-wave radar point-clouds,” *IEEE Access*, May 2021.

[4] P. Zhao *et al.*, “mID: Tracking and Identifying People with Millimeter Wave Radar,” in *Proc. DCOSS*, May 2019.

[5] 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. ACM mmNets*, Los Cabos, Mexico, 2019, pp. 51–56.

[6] D. Salami, R. Hasibi, S. Savazzi, T. Michoel, and S. Sigg, *Integrating sensing and communication in cellular networks via nr sidelink*, 2021.

[7] M. Ulrich, T. Hess, S. Abdulatif, and B. Yang, “Person recognition based on micro-doppler and thermal infrared camera fusion for firefighting,” in *21st International Conference on Information Fusion (FUSION)*, IEEE, 2018, pp. 919–926.

[8] 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 *2020 IEEE Radar Conference (RadarConf20)*, IEEE, Florence, Italy, Sep. 2020.

[9] S. M. Patole *et al.*, “Automotive radars: A review of signal processing techniques,” Mar. 2017.

[10] N. Knudde *et al.*, “Indoor tracking of multiple persons with a 77 GHz MIMO FMCW radar,” in *Proc. EURAD*, Oct. 2017.

[11] M. A. Richards, J. Scheer, W. A. Holm, and W. L. Melvin, *Principles of modern radar.* Scitech Publishing Inc., 2010.

[12] 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 Conference (RadarConf22)*, New York City, NY, USA, Mar. 2022.

[13] E. Bashirov, M. Canil, and M. Rossi, “RadNet: A Testbed for Mmwave Radar Networks,” in *Proc. ACM International Workshop on Emerging Topics in Wireless (EmergingWireless ’22)*, Rome, Italy, Dec. 2022.

[14] M. Canil, J. Pegoraro, A. Shastri, P. Casari, and M. Rossi, "Oracle: Occlusion-resilient and self-calibrating mmwave radar network for people tracking," *IEEE Sensors Journal*, pp. 1–1, 2023.

[15] M. Canil, J. Pegoraro, and M. Rossi, "milliTRACE-IR: Contact Tracing and Temperature Screening via mmWave and Infrared Sensing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 2, pp. 208–223, 2022.

[16] M. Canil, J. Pegoraro, J. O. Lacruz, *et al.*, "An experimental prototype for multistatic asynchronous isac," in *Proceedings of the First ACM Workshop on MmWave Sensing Systems and Applications*, ser. mmWave '23, Istanbul, Turkiye: Association for Computing Machinery, 2023, pp. 16–17.

[17] P. Vaishnav and A. Santra, "Continuous Human Activity Classification With Unscented Kalman Filter Tracking Using FMCW Radar," *IEEE Sensors Lett.*, Apr. 2020.

[18] R. G. Guendel *et al.*, "Continuous human activity recognition for arbitrary directions with distributed radars," in *Proc. IEEE RadarConf21*, May 2021.

[19] S. A. Shah and F. Fioranelli, "RF sensing technologies for assisted daily living in healthcare: A comprehensive review," Nov. 2019.

[20] S. Li *et al.*, "Pedestrian trajectory based calibration for multi-radar network," in *Proc. IEEE INFOCOM WKSHPS*, May 2021.

[21] S. Iwata *et al.*, "Multiradar data fusion for respiratory measurement of multiple people," *arXiv preprint arXiv:2107.11525*, 2021.

[22] R. E. Kalman, "A new approach to linear filtering and prediction problems," *ASME Transactions, Journal of Basic Engineering*, 1960.

[23] O. Sorkine-Hornung and M. Rabinovich, "Least-squares rigid motion using SVD," *Computing*, 2017.

[24] Kuhn, Harold W, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, 1955.

[25] J. Pegoraro, M. Canil, A. Shastri, P. Casari, and M. Rossi, *Oracle: Occlusion-resilient and self-calibrating mmwave radar network for people tracking*, 2022.

[26] D. Gubelli, O. A. Krasnov, and O. Yarovyi, "Ray-tracing simulator for radar signals propagation in radar networks," in *Proc. of EuRAD*, Nuremberg, Germany, 2013, pp. 73–76.

[27] S. Sundar Ram and H. Ling, "Simulation of human microdopplers using computer animation data," in *Proc. of IEEE RadarConf*, Rome, Italy, 2008, pp. 1–6.

[28] A. D. Singh, S. S. Ram, and S. Vishwakarma, "Simulation of the radar cross-section of dynamic human motions using virtual reality data and ray tracing," in *Proc. of IEEE RadarConf*, Oklahoma City, USA, 2018, pp. 1555–1560.

[29] Y. Deep, P. Held, S. S. Ram, *et al.*, "Radar cross-sections of pedestrians at automotive radar frequencies using ray tracing and point scatterer modelling," *IET Radar, Sonar Navig.*, vol. 14, 833–844(11), 6 Jun. 2020.

[30] A. Berson, *Client/Server Architecture (2nd Ed.)* USA: McGraw-Hill, Inc., 1996.

[31] B. A. Forouzan and S. C. Fegan, *TCP/IP Protocol Suite*, 2nd. McGraw-Hill Higher Education, 2002.

[32] A. Shastri, N. Valecha, E. Bashirov, *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, 2022.

[33] Z. Meng *et al.*, "Gait Recognition for Co-Existing Multiple People Using Millimeter Wave Sensing," in *Proc. AAAI*, Feb. 2020.

[34] M. Ester *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *2nd International Conference on Knowledge Discovery and Data Mining*, Aug. 1996.

[35] T. Wagner *et al.*, "Radar signal processing for jointly estimating tracks and micro-Doppler signatures," *IEEE Access*, Feb. 2017.

[36] C.-Y. Chong, S. Mori, W. 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, 2000.

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

[38] K. Bernardin, A. Elbs, and R. Stiefelhagen, "Multiple object tracking performance metrics and evaluation in a smart room environment," in *Proc. of IEEE Int. Wkshp. on Vis. Surveillance*, vol. 90, 2006.

[39] 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 Journal of Biomedical and Health Informatics*, 2023.

[40] M. J. Bocus and R. J. Piechocki, "Passive unsupervised localization and tracking using a multi-static UWB radar network," in *2021 IEEE Global Communications Conference (GLOBECOM)*, IEEE, Dec. 2021.

[41] Z. Xie and F. Ye, "Self-calibrating indoor trajectory tracking system using distributed monostatic radars for large scale deployment," in *Proc. ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys)*, Boston, MA, USA, Nov. 2022.

[42] V. S. Chernyak, *Fundamentals of multisite radar systems: multistatic radars and multistatic radar systems.* CRC press, 1998.

[43] 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 Conference (RadarConf16)*, Philadelphia, PA, USA, 2016.

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

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

[46] Y. He, F. Le Chevalier, and A. G. Yarovoy, "Range-Doppler processing for indoor human tracking by multistatic ultra-wideband radar," in *IEEE International Radar Symposium*, Warsaw, Poland, May 2012.

[47] H. Shu and Q. Liang, "Data fusion in a multi-target radar sensor network," in *Proc. IEEE Radio and Wireless Symposium (RWS)*, Long Beach, CA, USA, Jan. 2007.

[48] F. Folster and H. Rohling, "Data association and tracking for automotive radar networks," *IEEE Journal in Intelligent transportation systems*, vol. 6, no. 4, pp. 370–377, Dec. 2005.

[49] M. Rykunov, E. De Greef, K. Aziz, A. Bourdoux, H. Sahli, *et al.*, "Multi-radar fusion for failure-tolerant vulnerable road users classification," in *2021 18th European Radar Conference (EuRAD)*, IEEE, London, UK, Apr. 2022.

[50] A. Ahmed, S. Zhang, and Y. D. Zhang, “Multi-target motion parameter estimation exploiting collaborative UAV network,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 4459–4463.

[51] Y. Bar-Shalom, F. Daum, and J. Huang, “The probabilistic data association filter,” *IEEE Control Systems Magazine*, vol. 29, no. 6, pp. 82–100, 2009.

[52] O. Sorkine-Hornung and M. Rabinovich, “Least-squares rigid motion using SVD,” *Computing*, no. 1, pp. 1–5, 2017.

[53] Y. Bar-Shalom, F. Daum, and J. Huang, “The probabilistic data association filter,” *IEEE Control Systems Magazine*, vol. 29, no. 6, pp. 82–100, 2009.

[54] A. E. Hoerl and R. W. Kennard, “Ridge regression: Biased estimation for nonorthogonal problems,” *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[55] D. Reid, “An algorithm for tracking multiple targets,” *IEEE transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[56] C. T. Nguyen, Y. M. Saputra, N. Van Huynh, *et al.*, “Enabling and emerging technologies for social distancing: A comprehensive survey,” *arXiv preprint arXiv:2005.02816*, 2020.

[57] T. P. B. Thu, P. N. H. Ngoc, N. M. Hai, *et al.*, “Effect of the social distancing measures on the spread of COVID-19 in 10 highly infected countries,” *Science of The Total Environment*, vol. 742, p. 140 430, 2020.

[58] F. de Laval, A. Grosset-Janin, F. Delon, *et al.*, “Lessons learned from the investigation of a COVID-19 cluster in Creil, France: effectiveness of targeting symptomatic cases and conducting contact tracing around them,” *BMC Infectious Diseases*, vol. 21, no. 1, pp. 1–9, 2021.

[59] A. S. Ali and Z. F. Zaaba, “A study on contact tracing apps for covid-19: Privacy and security perspective,” *Webology*, vol. 18, no. 1, 2021.

[60] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[61] A. Farhadi and J. Redmon, “Yolov3: An incremental improvement,” *Computer Vision and Pattern Recognition*, 2018.

[62] M. I. Ribeiro, “Kalman and extended kalman filters: Concept, derivation and properties,” *Institute for Systems and Robotics*, vol. 43, p. 46, 2004.

[63] S. Savazzi, V. Rampa, L. Costa, S. Kianoush, and D. Tolochenko, “Processing of body-induced thermal signatures for physical distancing and temperature screening,” Early Access 2020.

[64] R. Faragher and R. Harle, “Location fingerprinting with bluetooth low energy beacons,” *IEEE journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2418–2428, 2015.

[65] I. Galvan-Tejada, E. I. Sandoval, R. Brena, *et al.*, “Wifi bluetooth based combined positioning algorithm,” *Procedia Engineering*, vol. 35, pp. 101–108, 2012.

[66] M. Cristani, A. Del Bue, V. Murino, F. Setti, and A. Vinciarelli, “The visual social distancing problem,” *IEEE Access*, vol. 8, pp. 126 876–126 886, 2020.

[67] S. Bian, B. Zhou, H. Bello, and P. Lukowicz, “A wearable magnetic field based proximity sensing system for monitoring COVID-19 social distancing,” in *Proceedings of the 2020 International Symposium on Wearable Computers*, 2020, pp. 22–26.

[68] M. Rezaei and M. Azarmi, “Deepsocial: Social distancing monitoring and infection risk assessment in covid-19 pandemic,” *Applied Sciences*, vol. 10, no. 21, p. 7514, 2020.

[69] A. J. Sathyamoorthy, U. Patel, Y. A. Savle, M. Paul, and D. Manocha, “Covid-robot: Monitoring social distancing constraints in crowded scenarios,” *arXiv preprint arXiv:2008.06585*, 2020.

[70] G. B. Dell’Isola, E. Cosentini, L. Canale, G. Ficco, and M. Dell’Isola, “Noncontact Body Temperature Measurement: Uncertainty Evaluation and Screening Decision Rule to Prevent the Spread of COVID-19,” *Sensors*, vol. 21, no. 2, p. 346, 2021.

[71] C. Ferrari, L. Berlincioni, M. Bertini, and A. Del Bimbo, “Inner eye canthus localization for human body temperature screening,” in *2020 25th International Conference on Pattern Recognition (ICPR)*, IEEE, 2021, pp. 8833–8840.

[72] T. Lewicki and K. Liu, “AI thermometer for temperature screening: demo abstract,” in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 2020, pp. 597–598.

[73] R. Zhang and S. Cao, “Extending reliability of mmwave radar tracking and detection via fusion with camera,” *IEEE Access*, Sep. 2019.

[74] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp, “A deep learning-based radar and camera sensor fusion architecture for object detection,” in *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, IEEE, 2019, pp. 1–7.

[75] B. Vandersmissen, N. Knudde, A. Jalalvand, *et al.*, “Indoor person identification using a low-power FMCW radar,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 7, pp. 3941–3952, Jul. 2018.

[76] M. Ye, J. Shen, G. Lin, T. Xiang, L. Shao, and S. C. Hoi, “Deep learning for person re-identification: A survey and outlook,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Early Access 2021.

[77] L. Fan, T. Li, R. Fang, R. Hristov, Y. Yuan, and D. Katabi, “Learning longterm representations for person re-identification using radio signals,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2020.

[78] Y. Cheng and Y. Liu, “Person reidentification based on automotive radar point clouds,” *IEEE Transactions on Geoscience and Remote Sensing*, Early Access 2021.

[79] J. Pegoraro and M. Rossi, “Real-time People Tracking and Identification from Sparse mm-Wave Radar Point-clouds,” 2021. arXiv: `2105.11368 [eess.SP]`.

[80] “How to find the right Thermal imaging camera,” *DIAS Infrared GmbH*, https://www.dias-infrared.de/pdf/How-to-find-the-right-thermal-imaging-camera_DIAS-Infrared.pdf, 2020.

[81] S. Prince, *Computer Vision: Models Learning and Inference.* Cambridge University Press, 2012.

[82] W. Zong, G.-B. Huang, and Y. Chen, “Weighted extreme learning machine for imbalance learning,” *Neurocomputing*, vol. 101, pp. 229–242, Feb. 2013.

[83] J. Nocedal and S. Wright, *Numerical optimization.* Springer Science & Business Media, 2006.

[84] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches.* John Wiley & Sons, 2006.

[85] L. Feng, S. Du, Z. Meng, A. Zhou, and H. Ma, “Evaluating mmWave Sensing Ability of Recognizing Multi-people Under Practical Scenarios,” in *International Conference on Green, Pervasive, and Cloud Computing*, Springer, Dec. 2020, pp. 61–74.

[86] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* MIT press, 2016.

[87] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[88] A. van den Oord, S. Dieleman, H. Zen, *et al.*, "WaveNet: A Generative Model for Raw Audio," in *The 9th ISCA Speech Synthesis Workshop*, Sep. 2016.

[89] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European conference on computer vision*, Springer, Oct. 2016.

[90] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE conference on computer vision and pattern recognition (CVPR)*, 2015.

[91] H. T. Huynh and Y. Won, "Regularized online sequential learning algorithm for single-hidden layer feedforward neural networks," *Pattern Recognition Letters*, vol. 32, no. 14, pp. 1930–1935, Oct. 2011.

[92] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[93] P. Viola, M. Jones, *et al.*, "Robust real-time object detection," *International journal of computer vision*, vol. 4, no. 34-47, p. 4, Feb. 2001.

[94] Z. Han, L. Han, X. Zhang, *et al.*, "Multistatic Integrated Sensing and Communication System in Cellular Networks," *arXiv preprint arXiv:2305.12994*, 2023.

[95] J. A. Zhang, K. Wu, X. Huang, Y. J. Guo, D. Zhang, and R. W. Heath, "Integration of radar sensing into communications with asynchronous transceivers," *IEEE Communications Magazine*, vol. 60, no. 11, pp. 106–112, 2022.

[96] J. Pegoraro, J. O. Lacruz, T. Azzino, *et al.*, "Jump: Joint communication and sensing with unsynchronized transceivers made practical," *arXiv preprint arXiv:2304.07766*, 2023.

[97] J. O. Lacruz, R. R. Ortiz, and J. Widmer, "A real-time experimentation platform for sub-6 ghz and millimeter-wave mimo systems," ser. MobiSys '21, Virtual Event, Wisconsin: Association for Computing Machinery, 2021, pp. 427–439.

List of publications

Journals

[14] M. Canil, J. Pegoraro, A. Shastri, P. Casari, and M. Rossi, “Oracle: Occlusion-resilient and self-calibrating mmwave radar network for people tracking,” *IEEE Sensors Journal*, pp. 1–1, 2023.

[15] M. Canil, J. Pegoraro, and M. Rossi, “milliTRACE-IR: Contact Tracing and Temperature Screening via mmWave and Infrared Sensing,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 2, pp. 208–223, 2022.

Conferences

[12] 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 Conference (RadarConf22)*, New York City, NY, USA, Mar. 2022.

[13] E. Bashirov, M. Canil, and M. Rossi, “RadNet: A Testbed for Mmwave Radar Networks,” in *Proc. ACM International Workshop on Emerging Topics in Wireless (EmergingWireless '22)*, Rome, Italy, Dec. 2022.

[16] M. Canil, J. Pegoraro, J. O. Lacruz, *et al.*, “An experimental prototype for multistatic asynchronous isac,” in *Proceedings of the First ACM Workshop on MmWave Sensing Systems and Applications*, ser. mmWave '23, Istanbul, Turkiye: Association for Computing Machinery, 2023, pp. 16–17.

Patent Applications

[98] A. Shastri, M. Canil, J. Pegoraro, P. Casari, and M. Rossi, “Method for self-calibration of mmWave radar networks,” in *Italian Patent Application*, no. 812022000069836, Mar. 2022.