Detection Layer Parameter Tuning Guide for the 3D People Tracking Demo

Revision 3.1

May 2023

Texas Instruments, Incorporated
12500 TI Boulevard
Dallas, Texas 75243 USA

Version History

Date	Comment
February, 2021	Rev1.0: – Initial version.
October, 2021	Rev2.0: – Two new commands (fineMotionCfg and bpmCfg) are added to the configuration parameters list. The corresponding sections of the document are updated to provide the tuning guide for these new commands. – Antenna phase rotations of AOP are updated to match with the given chirp configurations.
March, 2022	Rev3.0: – New configuration parameters are added to the fine motion mode command (fineMotionCfg). The corresponding sections of the document are updated to provide the tuning guide for these new parameters. – The processing scheme of the fine motion mode is detailed.
May, 2023	Rev3.1: – Renamed document to Detection Layer Parameter Tuning Guide ofr the 3D People Tracking Demo – Added new functionality to the <dynamicFlag> for the dynamic CFAR behavior

Table of Contents

1. Introduction and Scope

This document provides a high-level overview of the signal processing chain at the detection layer (discussed in the following subsection) for the three-dimensional (3D) people tracking demo. It also presents a guide for the user to tune the performance of this demo for different use-cases and environments. It is important to note that the scope of this document is limited to the detection layer only. It aims to explain the configurable detection layer parameters, which the user can tune to improve the overall demo performance. The low-level implementation details of the detection layer, including the execution flow, memory management, and benchmark analysis, can be found in [1]. The tracker layer's implementation details and parameter tuning guidelines are discussed in [2] and [3], respectively.

1.1. Detection Layer

In the 3D people tracking demo, the detection layer is responsible for sensing multiple reflections from the 3D environment around the sensor and delivering a rich set of measurement vectors, known as the point cloud, representing the real-life targets in the scene. The complex analog-to-digital converter (ADC) data (i.e., the beat signal) from the sensor front-end is the input of the detection layer. Each measurement vector generated by the detection layer represents a reflection point with range, azimuth, elevation, radial velocity (i.e., Doppler), and signal-to-noise ratio (SNR), as discussed in the following sections. A tracking layer [2][3] then processes the point cloud to localize and track the detected targets in the scene. A classification layer (**which is only part of the IWRL6432 people tracking demo**) can further classify the detected targets to create a complete processing chain to be visualized.

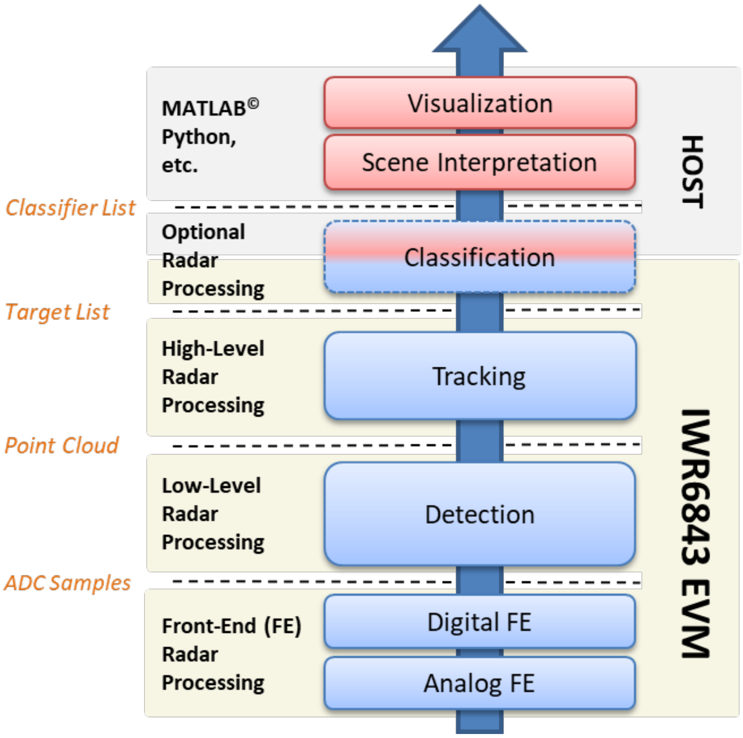


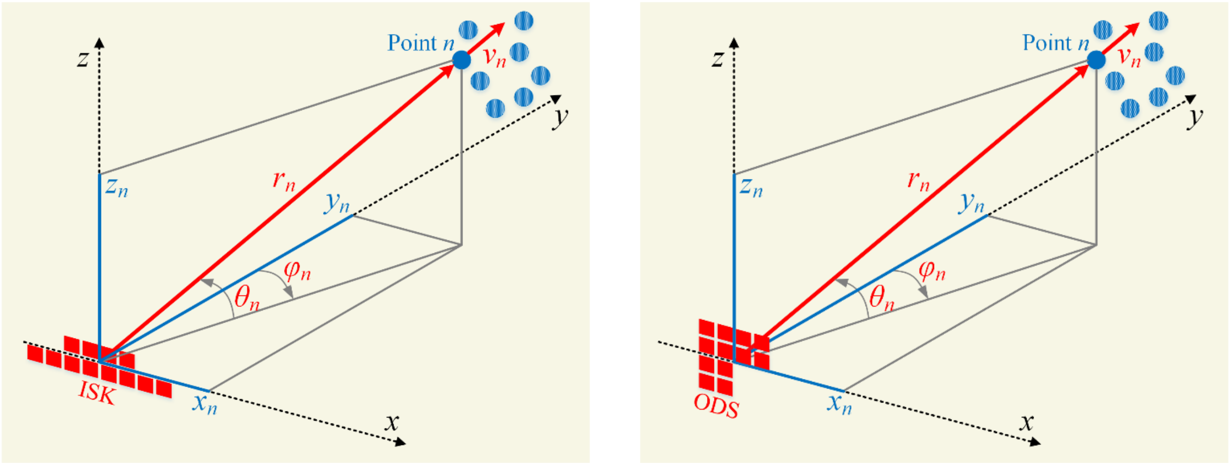
Figure 1. The signal processing layers of the 3D people tracking demo.

The flow diagram in Figure 1 illustrates the processing layers of the 3D people tracking demo. We develop this demo to run on the TI's IWR6843 mmWave radar sensors. As depicted in Figure 1, the existing processing chain running on the system-on-chips (SoC) provides the point cloud and the tracker information to the host that visualizes the complete data set to the user. In some use-cases, it may also be important to discriminate between humans moving through a scene and other movements generated by non-human motion sources. The optional classification layer depicted in Figure 1, which is not yet implemented in the current demo, can be developed to run either on the SoC or host to perform the object classification tasks.

1.2. System Geometry and Output Format

The detection layer running on the IWR6843 mmWave sensors supports both ISK and ODS/AOP type of antenna patterns (please refer to Section 3.3 for details). Utilizing the two-dimensional (2D) geometry of both antenna patterns, the detection layer outputs the 3D point cloud in spherical coordinates referenced to the sensor positioned at the established 3D Cartesian coordinate system's origin. The picture in Figure 2 illustrates the system geometry of a single reflection point *n* for both ISK and ODS antenna patterns (AOP antenna pattern has a similar layout with ODS, as discussed in Section 3.3). Multiple reflection points in the same geometry represent real-life radar targets. Each generated point from the target is represented by a range, angle (both azimuth and elevation), radial velocity, and SNR:

- Range *r*, $r_{min} < r < r_{max}$
- Azimuth angle φ, $-\emptyset_{max} < \varphi < +\emptyset_{max}$
- Elevation angle θ, $-\theta_{max} < \theta < +\theta_{max}$
- Radial velocity v, $-v_{max} < v < v_{max}$
- SNR



(a) (b)

Figure 2. The system geometry of the point cloud in the 3D coordinate system for (a) ISK and (b) ODS antenna patterns (see Section 3.3 for the antenna pattern details).

In Figure 2, the azimuth (φ) is defined as the angle from the y-axis to the orthogonal projection of the position vector onto the xy-plane. The angle is positive, going from the positive y-axis toward the positive x-axis. The elevation (θ) is defined as the angle from the projection onto the xy-plane to the vector. The angle is positive, going from the xy-plane to the positive z-axis.

It is important to note that the coordinate systems in Figure 2 are anchored to the radar sensors. In other words, the location of the sensor contains the origin of its local coordinate system. The detected points in the established sensor coordinate system follow these axes conventions:

- The positive x-axis points to the right, as viewed when facing forward,
- The positive y-axis points forward from the sensor (i.e., boresight),
- The positive z-axis points up from the xy-plane to maintain the right-handed coordinate system.

The 3D people tracking demo supports two sensor mount configurations (a) wall-mount and (b) ceil-mount. To interpret the detected point cloud in the global (i.e., world) coordinate system, where the real-world target information is generated, the user needs to understand how the radar is mounted in the global frame. The sensor mounting frame in Figure 2 will have some offset by a displacement from the global frame origin. Moreover, the sensor mounting frame may also have an angle rotation (i.e., azimuth or elevation tilt) in the global frame. The tracking and scene interpretation layers in Figure 1 handle the conversion step from local coordinates to global coordinates.

Figure 3 illustrates the relationship of local and global coordinate systems in both wall and ceil mount scenarios. The thick blue solid lines represent the global frame's coordinate axes, while both ISK (in wall-mount configuration) and ODS (in ceil-mount configuration) boards carry their local coordinate system depicted with dashed black lines. The user-configurable displacement and rotation parameters in both wall and ceil mount scenarios are discussed along with the coordinate transformation process in the tracker documentation [2][3].

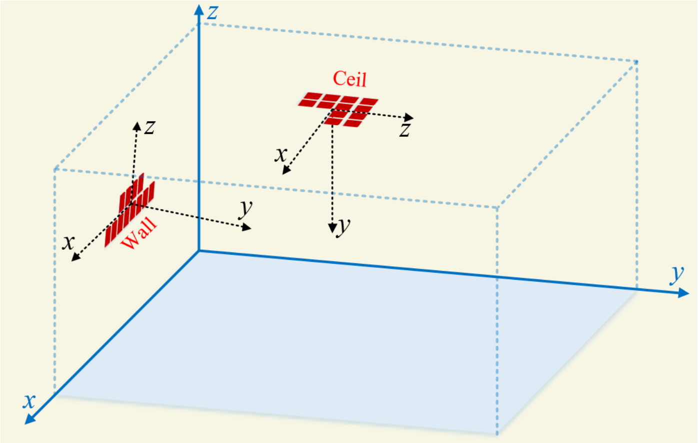


Figure 3. The global and local coordinate system for wall and ceil mount scenarios.

SNR in the generated point cloud is defined as the ratio of reflected signal power to the noise power in the linear scale. A ratio higher than 1:1 (i.e., greater than 0 decibels (dB)) indicates more signal than noise. In the radar domain, SNR is usually a function of the target's radar cross-section (RCS), and a larger SNR from a highly reflected object results in higher measurement accuracy. Please refer to the mmWave training series in [4] for the details about SNR.

2. Detection Algorithm

This section focuses on the high-level overview of the 3D people tracking algorithms for different scenarios and presents each processing block of these algorithms in detail. As discussed in Section 1.2 and detailed in the following subsections, there are two separate signal processing chains for the 3D people tracking demo called wall-mount and ceil-mount. This section presents the main blocks in both chains and the relation of each processing element with the scenario configurations.

2.1. 3D Wall-Mount Signal Processing Chain

This section presents the main signal processing blocks and the algorithm flow of the 3D wall-mount people tracking demo by focusing on the detection layer. The algorithm flows through the major steps depicted in the block diagram in Figure 4.

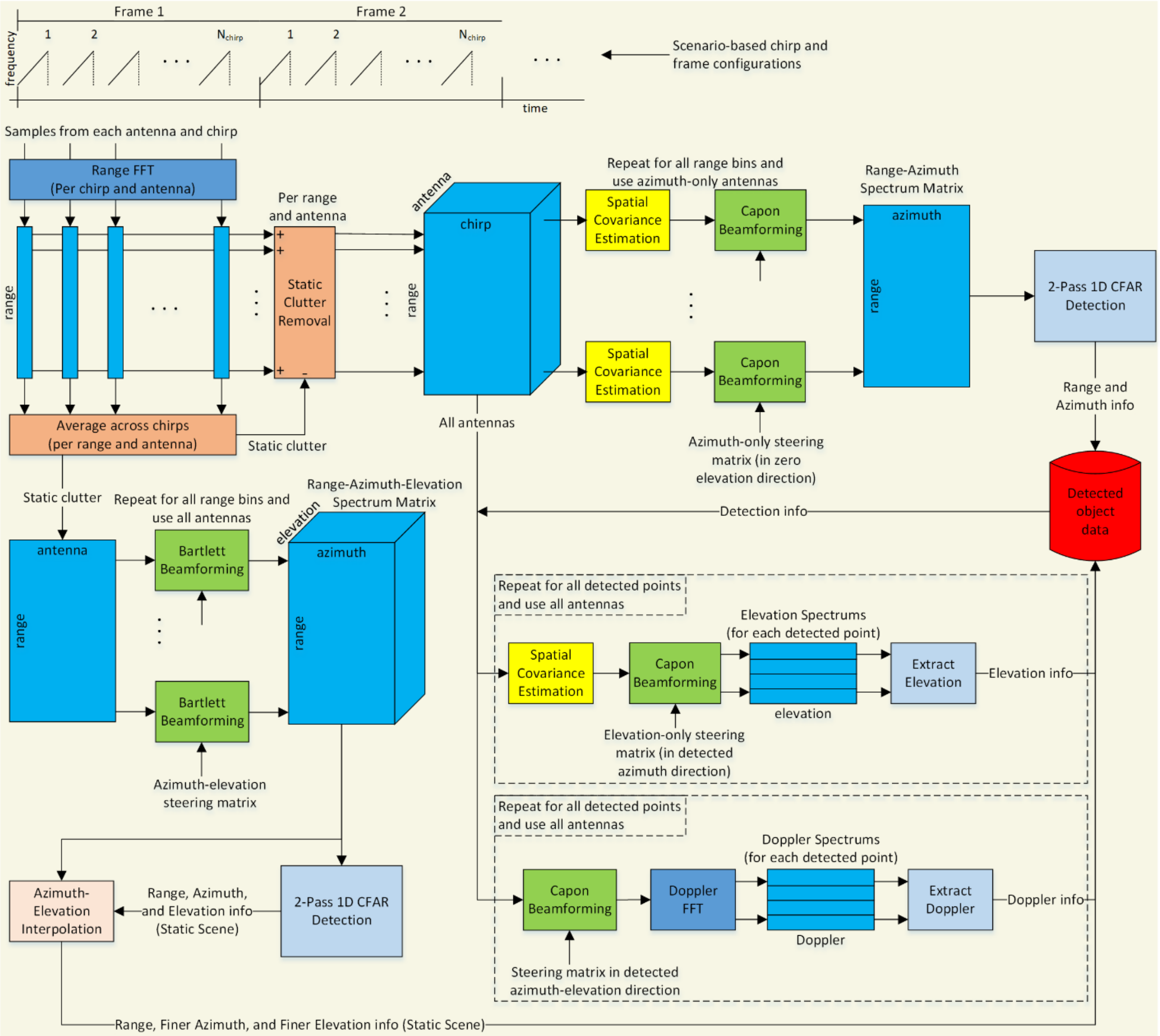

Frame 1
Frame 2
frequency
1
2
N_{chirp}
time
Scenario-based chirp and frame configurations
Samples from each antenna and chirp
Range FFT (Per chirp and antenna)
range
Per range and antenna
+
+
+
-
Static Clutter Removal
Average across chirps (per range and antenna)
Static clutter
antenna
chirp
All antennas
Repeat for all range bins and use azimuth-only antennas
Spatial Covariance Estimation
Capon Beamforming
Azimuth-only steering matrix (in zero elevation direction)
Range-Azimuth Spectrum Matrix
azimuth
2-Pass 1D CFAR Detection
Range and Azimuth info
Detected object data
Detection info
Static clutter
Repeat for all range bins and use all antennas
Bartlett Beamforming
Azimuth-elevation steering matrix
Range-Azimuth-Elevation Spectrum Matrix
elevation
Repeat for all detected points and use all antennas
Elevation Spectrums (for each detected point)
Elevation-only steering matrix (in detected azimuth direction)
Extract Elevation
Elevation info
Doppler Spectrums (for each detected point)
Doppler FFT
Doppler
Extract Doppler
Doppler info
Steering matrix in detected azimuth-elevation direction
Azimuth-Elevation Interpolation
Range, Azimuth, and Elevation info (Static Scene)
Range, Finer Azimuth, and Finer Elevation info (Static Scene)

Figure 4. The signal processing chain of the 3D wall-mount demo's detection layer.

Each processing step depicted in Figure 4 is explained briefly in the following subsections. The implementation details of each step are discussed in [1]. The configuration options of these steps are discussed in Section 3.2. In summary, for the given radar frame (see Section 3.1.3 for frame-based processing architecture), there is a list of detected points, where each point has a range, angle of arrival (azimuth and elevation), Doppler, and SNR value. The detection layer's output, which is the so-called point cloud, is used by the group tracker for object tracking [2][3].

2.1.1. Range Processing

The range processing step (i.e., windowing and fast-Fourier transform (FFT)) is applied over the ADC samples of each chirp from each TX-RX pair to create the range spectrum of the target scene.

2.1.2. Static Clutter Removal

The static clutter removal step is applied to remove the purely static objects such as chairs and tables from the scene and leave only the signals backscattered from the moving objects.

2.1.3. Capon Beamforming to Generate the Range-Azimuth Heatmap

After the range processing and static clutter removal steps, the Capon beamformer based on the steering vectors over a fine 1D azimuth grid (i.e., zero-elevation) is applied to generate the 2D range-azimuth spectrum matrix (i.e., heatmap). In the Capon beamformer, the azimuth-only transceiver pairs (TX-RX) are utilized to estimate the covariance matrix across the received chirps per range bin. The details about the antenna geometries of the radar boards are given in Section 3.3.1.

2.1.4. CFAR Detection for Range-Azimuth Estimation

In this step, a 2-pass constant false alarm rate (CFAR) algorithm is applied to the 2D range-azimuth heatmap for detection. In this approach, the detected points created by the first-pass CFAR-CASO (cell averaging smallest of) in the range domain are confirmed by a second-pass search step in the azimuth domain. The second-pass search is either a CFAR-CASO or a local peak search based on a configurable logic to confirm the detected points created by the first-pass range-CFARs.

2.1.5. Elevation Estimation

For each detected point in the range-azimuth domain after the 2-pass CFAR, a Capon beamforming algorithm is applied to generate the 1D elevation spectrum. The strongest (i.e., maximum) peak in the elevation spectrum is then selected as the elevation angle of the detected point.

2.1.6. Doppler Estimation

A Capon beamformer pointing to the estimated azimuth-elevation angle is applied to the virtual antennas per detected range. The Doppler spectrum is then extracted over the consecutive chirps in the radar data cube (see Section 3.1.3 for details), followed by a maximum peak search in the Doppler spectrum to estimate the radial velocity of the detected point.

2.1.7. Static Scene Processing

In addition to the dynamic points (i.e., people, etc.), the static scene reflections can also be detected in the 3D wall-mount people tracking demo using the conventional Bartlett beamforming approach followed by a 2-pass CFAR step (discussed in Section 2.1.4).

2.2. 3D Ceil-Mount Signal Processing Chain

This section presents the main signal processing blocks and the algorithm flow of the 3D ceil-mount people tracking demo by focusing on the detection layer. The algorithm flows through the major steps depicted in the block diagram in Figure 5.

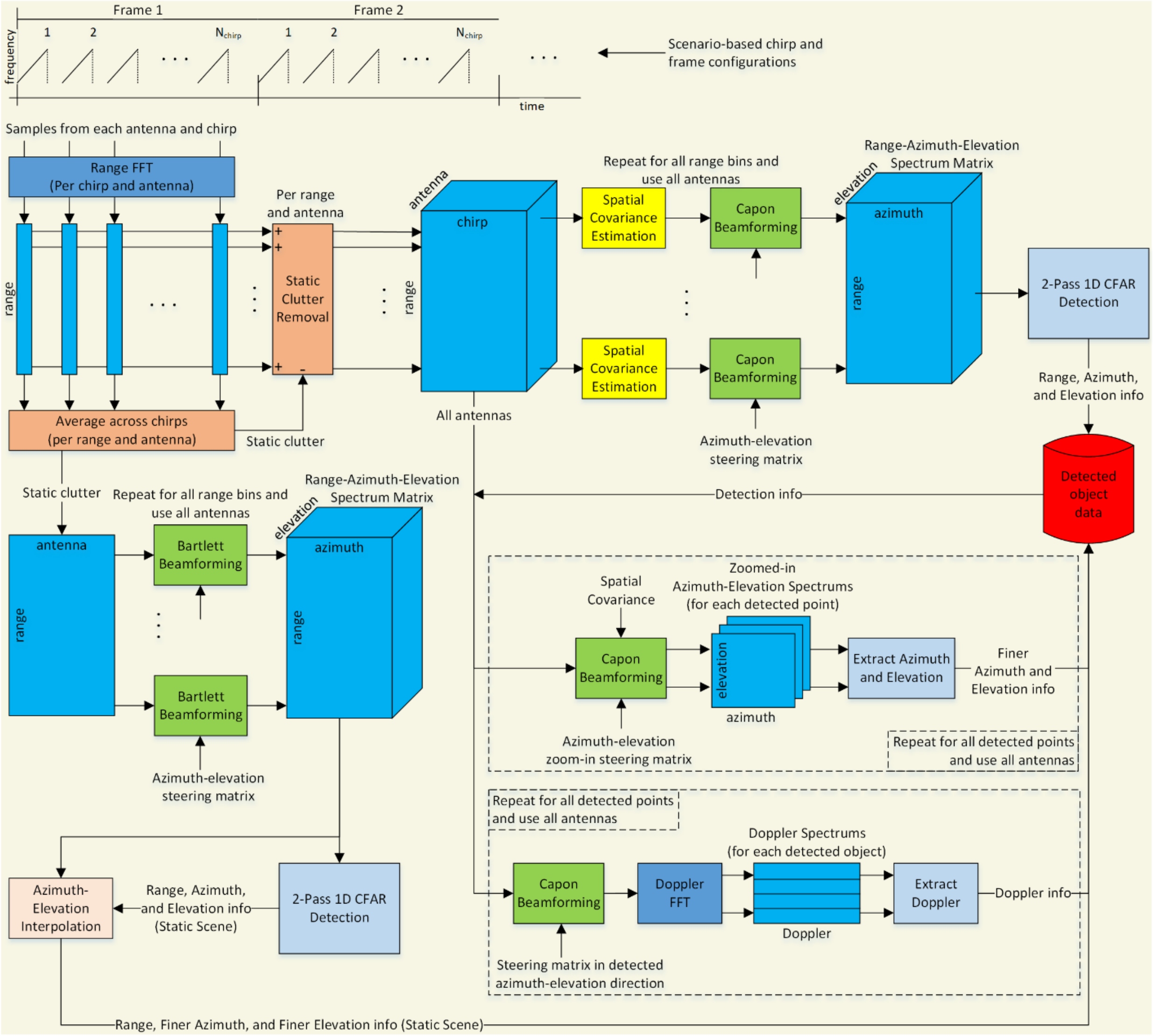

Frame 1
Frame 2
frequency
1
2
N_{chirp}
1
2
N_{chirp}
. . .
time
Scenario-based chirp and frame configurations
Samples from each antenna and chirp
Range FFT (Per chirp and antenna)
range
Per range and antenna
Static Clutter Removal
+
-
antenna
chirp
range
All antennas
Average across chirps (per range and antenna)
Static clutter
Repeat for all range bins and use all antennas
Spatial Covariance Estimation
Capon Beamforming
Spatial Covariance Estimation
Capon Beamforming
Azimuth-elevation steering matrix
Range-Azimuth-Elevation Spectrum Matrix
elevation
azimuth
range
2-Pass 1D CFAR Detection
Range, Azimuth, and Elevation info
Detected object data
Detection info
Static clutter
Repeat for all range bins and use all antennas
antenna
range
Bartlett Beamforming
Bartlett Beamforming
Azimuth-elevation steering matrix
Range-Azimuth-Elevation Spectrum Matrix
elevation
azimuth
range
Spatial Covariance
Zoomed-in Azimuth-Elevation Spectrums (for each detected point)
Capon Beamforming
elevation
azimuth
Extract Azimuth and Elevation
Finer Azimuth and Elevation info
Azimuth-elevation zoom-in steering matrix
Repeat for all detected points and use all antennas
Repeat for all detected points and use all antennas
Doppler Spectrums (for each detected object)
Capon Beamforming
Doppler FFT
Doppler
Extract Doppler
Doppler info
Steering matrix in detected azimuth-elevation direction
Azimuth-Elevation Interpolation
Range, Azimuth, and Elevation info (Static Scene)
2-Pass 1D CFAR Detection
Range, Finer Azimuth, and Finer Elevation info (Static Scene)

Figure 5. The signal processing chain of the 3D ceil-mount demo's detection layer.

As illustrated in Figure 4 and Figure 5, the key difference between the wall-mount and ceil-mount configurations is the way of estimating the angle (i.e., azimuth and elevation) of each detected dynamic point. In wall-mount scenarios, an accurate azimuth angle estimation, which is the primary contributor to successfully tracking multiple people in the xy-plane (see Figure 3), is essential. Therefore, the wall-mount processing chain first applies the range-azimuth estimation step utilizing azimuth-only transceiver pairs and a fine azimuth grid in the Capon beamformer. The elevation of each detected point is then estimated using all the available antennas. In this step, the 1D elevation spectrum is created only for the detected azimuth bins, and a single peak is searched to find out the estimated elevation index. On the other hand, in ceil-mount scenarios, an accurate estimate of both azimuth and elevation angles is necessary to track multiple people in the xy-plane (see Figure 3). Hence, the ceil-mount processing chain constructs a full azimuth-elevation heatmap utilizing all the available antennas and runs a two-step estimation approach (coarse estimation followed by zoom-in processing) on the azimuth-elevation domain.

Similar to the wall-mount counterpart, each processing step of the 3D ceil-mount people tracking demo's detection layer is explained briefly in the following subsections. The implementation details of each step are discussed in [1]. The configuration options of these steps are discussed in Section 3.2. In summary, for the given radar frame, there is a list of detected points, where each point has a range, angle of arrival (azimuth and elevation), Doppler, and SNR value. The detection layer's output, which is the so-called point cloud, is used by the group tracker for object tracking [2][3].

2.2.1. Range Processing

The range processing step (i.e., windowing and FFT) is applied over the ADC samples of each chirp from each TX-RX pair to create the range spectrum of the target scene.

2.2.2. Static Clutter Removal

The static clutter removal step is applied to remove the purely static objects such as chairs and tables from the scene and leave only the signals backscattered from the moving objects.

2.2.3. Capon Beamforming to Generate the Range-Azimuth-Elevation Heatmap

After the range processing and static clutter removal steps, the Capon beamformer based on the steering vectors over a coarse 2D azimuth-elevation angular grid is applied to generate the 3D range-azimuth-elevation spectrum matrix (i.e., heatmap). Compared with the 3D wall-mount processing chain, all the available transceiver pairs (TX-RX) are used to estimate the covariance matrix across the received chirps per range bin. Due to the limited MIPS (million instructions per second) and memory resources, coarse angle steps are used in the azimuth-elevation grid, which can be configured by the user. The following finer azimuth-elevation estimation step is discussed in Section 2.2.5 to improve angle estimation accuracy.

2.2.4. CFAR Detection for Range-Azimuth-Elevation Estimation

Similar to the detection approach applied to the 3D wall-mount chain, a 2-pass CFAR algorithm is applied to the created 3D range-azimuth-elevation heatmap for detection. Before the CFAR detection, the generated 3D range-azimuth-elevation heatmap is first reshaped into a 2D range-angle heatmap with one dimension on the range and another dimension on the angle. In the reshaped version, azimuth-elevation angles are compressed into one dimension as (aziInd, eleInd) → (aziInd + eleInd * azimDim). The main reason behind this reshaping step is to create a 2D range-angle heatmap for the first-pass range-CFARs per angle bin. In the 2-pass CFAR algorithm applied to the reshaped 2D range-angle heatmap, a second-pass CFAR-CASO or a local peak search in the angle domain confirms the detection points created by the first-pass range-CFARs. Both the azimuth and elevation index are then extracted from the detected angle index.

2.2.5. Finer Azimuth-Elevation Estimation

After the 2-pass CFAR, the Capon beamforming algorithms based on the steering vectors over a finer 2D azimuth-elevation angular grid around the detected coarse azimuth-elevation points are applied to generate the zoomed-in 2D azimuth-elevation angle spectrums. The strongest peak of each zoomed-in spectrum and its neighbors that meet specific criteria are then selected as the detected points' finer azimuth and elevation angles.

2.2.6. Doppler Estimation

A Capon beamformer pointing to the estimated azimuth-elevation angle is applied to the virtual antennas per detected range. The Doppler spectrum is then extracted over the consecutive chirps in the radar data cube (see Section 3.1.3 for details), followed by a maximum peak search in the Doppler spectrum to estimate the radial velocity of the detected point.

2.2.7. Static Scene Processing

In addition to the dynamic points (i.e., people, etc.), the static scene reflections can also be detected in the 3D ceil-mount people tracking demo using the conventional Bartlett beamforming approach followed by a 2-pass CFAR step (discussed in Section 2.1.4).

3. Configuration Parameters

The parameters presented in this section are used to configure the people tracking demo and can be adjusted to match the use-cases based on the particular scenery and target characteristics. The user can set the configuration parameters using the configuration (i.e., cfg) files located in the corresponding people tracking demo (i.e., wall-mount or ceil-mount). Example configuration files for the 3D wall-mount and 3D ceil-mount people tracking demos are given in Table 1 and Table 2, respectively. The configuration file formats are the same for both the 3D wall and ceil-mount scenarios except for some differences in the parameter settings described in the following subsections. Each line in this file represents a command-line interface (CLI) message that configures several parameters.

Table 1. An example configuration (cfg) file for the 3D wall-mount people tracking demo.

No	Parameter group	CLI commands and arguments
1	Sensor front-end parameters	dfeDataOutputMode 1 channelCfg 15 7 0 adcCfg 2 1 adcbufCfg -1 0 1 1 1 lowPower 0 0 profileCfg 0 60.75 30 25 59.1 657930 0 54.71 1 96 2950 2 1 36 chirpCfg 0 0 0 0 0 0 0 1 chirpCfg 1 1 0 0 0 0 0 2 chirpCfg 2 2 0 0 0 0 0 4 frameCfg 0 2 96 0 55 1 0
2	Detection layer parameters	dynamicRangeAngleCfg -1 0.75 0.0010 1 0 dynamicRACfarCfg -1 4 4 2 2 8 12 4 8 5.00 8.00 0.40 1 1 dynamic2DAngleCfg -1 3 0.0300 1 0 1 0.50 0.85 8.00 staticRangeAngleCfg -1 0 8 2 staticRACfarCfg -1 6 2 2 2 8 8 6 4 8.00 15.00 0.30 0 0
3	Board related parameters	antGeometry0 0 -1 -2 -3 -2 -3 -4 -5 -4 -5 -6 -7 antGeometry1 -1 -1 -1 -1 0 0 0 0 -1 -1 -1 -1 antPhaseRot 1 1 1 1 1 1 1 1 1 1 1 1 fovCfg -1 70.0 20.0 compRangeBiasAndRxChanPhase 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
4	Tracking layer parameters	staticBoundaryBox -3 3 1 6 0 3 boundaryBox -4 4 0 8 0 3 sensorPosition 2 0 15 gatingParam 3 2 2 2 4 stateParam 3 3 6 500 5 6000 allocationParam 40 100 0.1 20 0.5 20 maxAcceleration 0.1 0.1 0.1 presenceBoundaryBox -4 4 -4 4 0.5 2.5 trackingCfg 1 2 800 20 46 96 55

Note that the commands listed in both Table 1 and Table 2 are all mandatory to run the 3D people tracking demo. In addition to these parameters, there are also some optional commands that the user can configure to improve the performance in some use cases. Both the mandatory and optional commands are detailed in the following subsections.

Table 2. An example configuration (cfg) file for the 3D ceil-mount people tracking demo.

No	Parameter group	CLI commands and arguments
1	Sensor front-end parameters	dfeDataOutputMode 1 channelCfg 15 7 0 adcCfg 2 1 adcbufCfg -1 0 1 1 1 lowPower 0 0 profileCfg 0 61.2 60.00 17.00 50 657930 0 55.27 1 64 2000.00 2 1 36 chirpCfg 0 0 0 0 0 0 0 1 chirpCfg 1 1 0 0 0 0 0 2 chirpCfg 2 2 0 0 0 0 0 4 frameCfg 0 2 224 0 120.00 1 0
2	Detection layer parameters	dynamicRangeAngleCfg -1 7.000 0.0010 2 0 dynamicRACfarCfg -1 10 1 1 1 8 8 6 4 4.00 6.00 0.50 1 1 dynamic2DAngleCfg -1 5 1 1 1.00 15.00 2 staticRangeAngleCfg -1 0 1 1 staticRACfarCfg -1 4 4 2 2 8 16 4 6 6.00 13.00 0.50 0 0
3	Board related parameters	antGeometry0 0 0 -1 -1 -2 -2 -3 -3 -2 -2 -3 -3 antGeometry1 0 -1 -1 0 0 -1 -1 0 -2 -3 -3 -2 antPhaseRot 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 fovCfg -1 60.0 60.0 compRangeBiasAndRxChanPhase 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
4	Tracking layer parameters	staticBoundaryBox -3 3 -3 3 -0.5 3 boundaryBox -4 4 -4 4 -0.5 3 sensorPosition 2.9 0 90 gatingParam 2 2 2 2 4 stateParam 3 3 6 300 5 1000 allocationParam 5 100 0.01 10 1.5 20 maxAcceleration 0.2 0.01 0.2 presenceBoundaryBox -3 3 2 6 0.5 2.5 trackingCfg 1 4 800 20 37 33 120

This document elaborates specifically on the configuration parameters unique to the detection layer of the 3D people tracking demo specified by range-angle heatmap generation, constant false alarm rate (CFAR) detection [1], azimuth-elevation estimation, etc., and the board related parameters specified by antenna pattern, field-of-view (FOV), calibration vector, etc., listed in the detection layer and board related parameter groups (i.e., group number 2 and 3) of the tables above.

The group tracker configuration parameters listed in the tracking layer parameter group (i.e., group number 4) are specified in [3]. Those parameters can control the initial target allocation behavior, subsequent tracking, and deallocation of the tracks.

The remaining configuration parameters for the low-level sensor front-end processing are detailed in the mmWave software development kit (SDK) [5]. In this document, we also briefly review these parameters and discuss their effect on the overall people tracking performance.

It is important to emphasize that we recommend using the provided values of the configuration parameters as the best empirical values we obtained from testing in different environments. In the following sections, we provide detailed information on each parameter and discuss their effect on the detection performance for different scenarios.

3.1. Sensor Front-End Parameters

The sensor front-end parameters, which configure the 3D people tracking demo at the frequency modulated continuous wave (FMCW) signal level, are described in the mmWave-SDK user guide [5] in detail. The following subsections briefly summarize these parameters at the 3D people tracking demo level to provide a complete guide for the user. To get a general understanding of the sensor front-end parameters, please refer to the mmWave-SDK user guide. Please note that this demo does not support all the configuration options listed in the user guide. The supported options are discussed in the following subsections.

3.1.1. Radar RF Configuration

The table below summarizes the commands to configure the RF module parameters of the radar sensor, such as enabling the transmit and receive antennas and configuring the ADC settings. The table also presents the default values of the 3D people tracking demo parameters and provides information regarding the supported features in the signal processing chain.

Commands	Parameters	Usage in 3D people tracking demo	Notes
dfeDataOutputMode	<modeType>	Always set to 1.	Only the frame-based chirps mode is supported.
channelCfg	<rxChannelEn>	Always set to 15.	All the receiving antennas should be switched-on using the mask 0x1111b = 15.
	<txChannelEn >	Always set to 7.	All the transmitting antennas should be switched-on using the mask 0x111b = 7.
	<cascading>	Always set to 0.	Cascading mode is not applicable.
adcCfg	<numADCBits>	Always set to 2.	Only 16-bit is supported.
	<adcOutputFmt>	Always set to 1.	Only complex 1x is supported.
adcbufCfg	<subFrameIdx>	Always set to -1.	Advanced frame mode is not supported.
	<adcOutputFmt>	Always set to 0.	Only the complex mode is supported.
	<sampleSwap>	Always set to 1.	Only Q in LSB and I in MSB option is supported.
	<chanInterleave>	Always set to 1.	Only the non-interleaved option is supported.
	<chirpThreshold>	Always set to 1.	People tracking demo uses HWA for the 1D range FFT.
lowPower	-	Always set to 0.	This parameter is not used.
	<adcMode>	Always set to 0.	Only the regular ADC mode is supported.

3.1.2. Chirp Profile Configuration

We develop the 3D people tracking signal processing chain to run on the TI's IWR6843 mmWave sensors that use the FMCW (also known as chirp) signal waveform. The basic principles of FMCW radars are well reported in the literature [7][8]. Here, we briefly review a single element FMCW radar signal model to recall the terminology used in this document.

In the complex baseband, the time-domain signal model of the generated beat (or intermediate frequency) signal, which consists of in-phase (I) and quadrature (Q) components, is defined as

$$s(t) = s_I(t) + s_Q(t) = \exp\big(j2\pi(f_0\tau + Kt\tau - 0.5K\tau^2)\big), \qquad 0 \le t \le T \tag{1}$$

where f_0 is the start frequency, $K = B/T$ is the slope of frequency computed from the valid sweep bandwidth of B, and the chirp duration of T. In (1), τ is the round-trip delay of the backscattered signal assuming a single ideal scatterer in the scene, and $K\tau$ term is the beat frequency which carries the range information of this scatterer.

The FMCW chirp configuration is an important performance aspect that needs to be considered in the 3D people tracking chain. The IWR6843 mmWave sensors provide flexibility in configuring the chirp waveforms through the profileCfg command. This section summarizes the parameters in this command that govern the chirp configurations optimized for the 3D people tracking demo. To establish the terminology used in this paper, Figure 6 illustrates the chirp signal model along with these parameters.

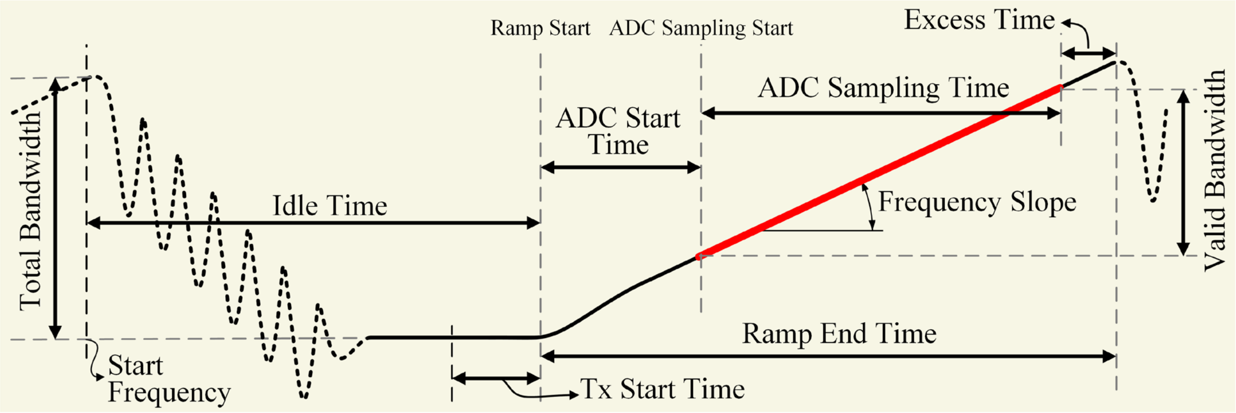


Figure 6. The FMCW chirp signal parameters.

It is important to note that the chirp duration T given in (1) refers to the ADC sampling time in Figure 6, which is computed by the number of ADC samples (numAdcSamples) and the sampling rate (digOutSampleRate) parameters of the profileCfg command detailed in this section.

In all the tables in this document that describe different command sets, the parameters are listed in **row-major order**. Each table presents the data type and description of each parameter along with the suggested values in **3D Wall-Mount (3DWM)** and **3D Ceil-Mount (3DCM)** scenarios (please refer to Appendix 6.1 for the template of the tables).

profileCfg			
<profileId>	<startFreq>	<idleTime>	<adcStartTime>
int32	float	float	float
0	3DWM : 60.75 3DCM : 61.2	3DWM : 30 3DCM : 60	3DWM : 25 3DCM : 17
Profile identifier. Always set to 0. Only a single profile is supported.	Start frequency in GHz. Refer to Figure 6, the report in [8], and the note below[1] for the details.	Idle time in µs. Refer to Figure 6, the report in [8], and the note below[2] for the details.	ADC start time in µs. Refer to Figure 6, the report in [8], and the note below[1] for the details.
<rampEndTime>	<txOutPowerBackoff>	<txPhaseShifter>	<freqSlope>
float	int32	int32	float
3DWM : 59.1 3DCM : 50	657930 (10dB)	0	3DWM : 54.71 3DCM : 55.27
Ramp end time in µs. Refer to Figure 6, the report in [8], and the description below[3] for the details.	TX output power backoff code. Refer to the detailed description below[4].	Phase shifter for the TX antennas. Always set to 0.	Frequency slope in MHz/µs. Refer to Figure 6, the report in [8], and the note below[1] for the details.
<txStartTime>	<numAdcSamples>	<digOutSampleRate>	<hpfCornerFreq1>
float	int32	int32	int32
1	3DWM : 96 3DCM : 64	3DWM : 2950 3DCM : 2000	2
TX start time in µs. Refer to Figure 6 and the report in [8] for the details.	Number of ADC samples collected during the ADC sampling time. Refer to the report in [8] for the details.	ADC sampling frequency in ksps. Refer to the report in [8] for the details.	High pass filter 1 corner frequency. Refer to [5] for the details.
<hpfCornerFreq2>	<rxGain>	-	-
int32	int32	-	-
1	36	-	-
High pass filter 2 corner frequency. Refer to [5] for the details.	RX gain in dB scale. Refer to [5] for the details.	-	-

[1]**startFreq:** As illustrated in Figure 6, this parameter is the start frequency of the total occupied bandwidth. Along with the freqSlope and adcStartTime, this parameter configures the valid bandwidth occupied in the raw FMCW beat signal to be utilized in the detection layer. The given values for the 3D wall-mount and ceil-mount chains are configured empirically for the best performance according to the radiation patterns of the ISK and ODS/AOP antennas, respectively. The user might need to adjust these values for their customized antenna designs.

[2]**idleTime:** As illustrated in Figure 6, this parameter is the idle time between two consecutive chirps. Along with the rampEndTime, the number of TX antennas, and the number of chirp loops in a frame (as discussed in Section 3.1.3), this parameter configures the motion sensitivity (i.e., velocity resolution) of the detection layer (please refer to Section 4 for details). Due to the smaller radial velocities observed in overhead scenarios, the velocity resolution of the 3D ceil-mount processing chain is configured as higher than the 3D wall-mount counterpart. The user can adjust this parameter according to desired motion sensitivity. It is important to note that the user must ensure that the total chirping time and the required processing time can fit into a single frame period, which is discussed in Section 3.1.3.

[3]**rampEndTime:** As illustrated in Figure 6, this parameter is the sum of adcStartTime, ADC sampling time, and excess time. Therefore, the user must ensure that the adcStartTime + ADC sampling time is less than the rampEndTime in profileCfg to keep at least 1µs excess time.

[4]**txOutPower:** This parameter is the concatenated code of output power backoff for TX0, TX1, TX2 according to the bit description given below. Each backoff code in this field defines how much the transmit power for each transmit antenna should be reduced from the maximum in dB scale. 0dB backoff corresponds to typically 12dBm power level for the IWR6843 mmWave sensors. This parameter should be configured according to the detection range requirements of the use-case and transmission requirements mandated by the regulations, which may differ in different regions of the world. The suggested value of 10dBm is empirically adjusted to achieve a better detection performance both for wall-mount and ceil-mount scenarios. Configuring the power as too high (i.e., using a backoff value close to 0dB) may cause stronger multipath reflections and higher false alarms. On the other hand, too low TX power will decrease the detection range performance (see Section 4), which may result in more missed detections.

Bit description of the TX output power backoff code is:

- b7:0 TX0 output power backoff in dB scale
- b15:8 TX1 output power backoff in dB scale
- b23:16 TX2 output power backoff in dB scale
- b31:24 Reserved

3.1.3. Frame Configuration

As illustrated in Figure 4 and Figure 5, both the wall-mount and ceil-mount people tracking demos utilize the 3D radar data cube as input similar to the other conventional radar signal processing chains. The first axis of the radar data cube corresponds to the range spectrum computed from time-domain ADC samples of each chirp signal (configured by the profileCfg command as detailed in Section 3.1.2). It is important to emphasize that the zero-padded range-FFT size is always the nearest next power of two of the number of ADC samples. The second axis of the radar cube holds the spatially sampled signals using multiple virtual antennas to estimate the direction-of-arrival (DOA) of the targets. The third axis (typically known as the slow-time domain) is used to store a set of chirps from each virtual channel to provide a basis for estimating the motion of the targets.

This section presents the configuration parameters that govern the second (i.e., the antenna domain) and third (i.e., the chirp domain) axis of the radar cube. Figure 7 illustrates the 3D radar data cube format and its generation process along with the radar cube sizes of both wall-mount (WM) and ceil-mount (CM) configurations (where each complex sample is 4 bytes in size).

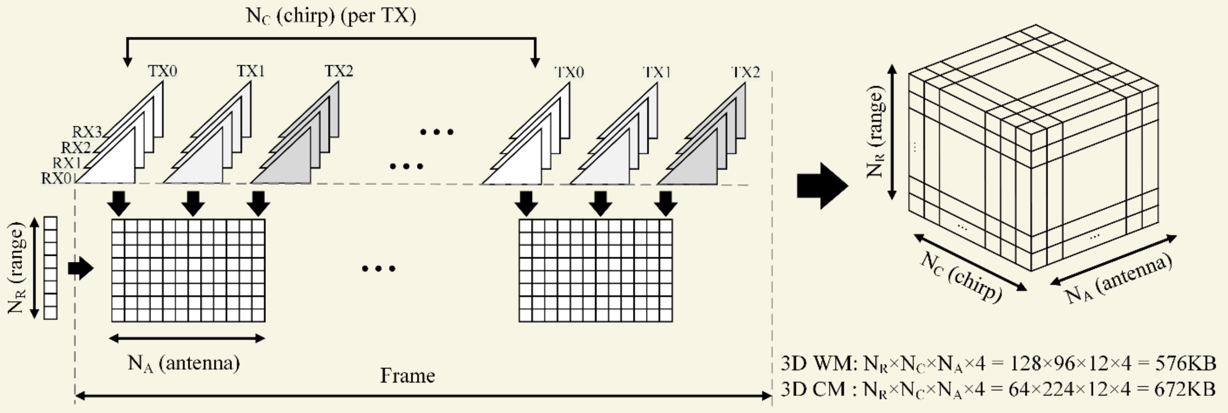

N_C (chirp) (per TX)
TX0
TX1
TX2
TX0
TX1
TX2
RX3
RX2
RX1
RX0
N_R (range)
N_A (antenna)
Frame
N_R (range)
N_C (chirp)
N_A (antenna)
3D WM: $N_R \times N_C \times N_A \times 4 = 128 \times 96 \times 12 \times 4 = 576$KB
3D CM : $N_R \times N_C \times N_A \times 4 = 64 \times 224 \times 12 \times 4 = 672$KB

Figure 7. Generation of the 3D radar data cube.

The IWR6843 mmWave sensors have three transmit and four receive antennas. To enable the MIMO mode on these sensors [7], the receiving elements must be able to separate the signals corresponding to different transmitter antennas. In the 3D people tracking demo, the orthogonality between the transmit antennas is achieved by employing both the time-division multiplexing (TDM) and binary phase modulation (BPM) techniques [7].

The following chirpCfg command configures both the TDM-MIMO and BPM-MIMO schemes by enabling the TX antennas in a particular sequence. As noted in the example configuration (cfg) files, this command must be run three times consecutively for each distinct TX antenna, which, in turn, creates the virtual antenna domain of the 3D radar data cube, as illustrated in Figure 7.

It is important to note that the 3D people tracking demo supports the BPM-MIMO scheme with only two TX antennas. In this mode, two TX antennas should transmit simultaneously in BPM mode, and the third TX antenna should be in TDM mode. To achieve the best performance in the 3D wall-mount processing chain (where this scheme is tested), the two TX antennas in BPM-MIMO mode should be in the same azimuth axis (for details, please refer to the antenna patterns in Section 3.3). For more details about the BPM mode, please refer to Section 3.2.7.

chirpCfg (Note: this command is run three times consecutively for each distinct TX antenna)			
<chirpStartIdx>	<chirpEndIdx>	<profileId>	<startFreqVar>
int32	int32	int32	float
0, 1, or 2	0, 1, or 2	0	0
Chirp start index. Always set to: ▪ 0 for the TX0 ▪ 1 for the TX1 ▪ 2 for the TX2	Chirp end index. Always set to: ▪ 0 for the TX0 ▪ 1 for the TX1 ▪ 2 for the TX2	Profile identifier. Always set to 0. As defined in the profileCfg->profileId, only a single profile is supported.	Start frequency variation in Hz. Suggested to set 0. Configure the start frequency in the profileCfg command.
<freqSlopeVar>	<idleTimeVar>	<adcStartTimeVar>	<txEnableMask>
float	float	float	int32
0	0	0	1, 2, or 4
Frequency slope variation in kHz/µs. Suggested to set 0. Configure the frequency slope in the profileCfg command.	Idle time variation in µsec. Suggested to set 0. Configure the idle time in the profileCfg command.	ADC start time variation in µsec. Suggested to set 0. Configure the ADC start time in the profileCfg command.	TX antenna enable mask. In TDM-MIMO mode, individual chirps should have only one distinct TX antenna enabled. In TDM-MIMO mode, always set to: ▪ (001)b = 1 for the TX0 ▪ (010)b = 2 for the TX1 ▪ (100)b = 4 for the TX2 In the supported BPM-MIMO mode, two TX antennas should transmit simultaneously in BPM mode, and the third TX antenna should be in TDM mode. In BPM-MIMO mode, always set (the two antennas in BPM mode) to: ▪ (011)b = 3 for the TX0 and TX1 ▪ (101)b = 5 for the TX0 and TX2 ▪ (110)b = 6 for the TX1 and TX2 In the supported BPM-MIMO mode, the third TX antenna should use one of the TDM antenna masks listed above.

The following command configures the total number of chirp loops in a single frame, which, in turn, creates the chirp domain of the 3D radar data cube, as illustrated in Figure 7.

frameCfg			
<chirpStartIdx>	<chirpEndIdx>	<numLoops>	<numFrames>
int32	int32	int32	int32
0	2	3DWM : 96 3DCM : 224	0
Chirp start index. Always set to 0 to include the chirps from TX0, TX1, and TX2.	Chirp end index. Always set to 2 to include the chirps from TX0, TX1, and TX2.	Number of loops in a single frame. This parameter loops the chirps of TX0->TX1->TX2 sequence numLoops times. Refer to the note below[1].	Number of total frames. Always set to 0 to run the demo in continuous mode.
<framePeriodicity>	<triggerSelect>	<frameTriggerDelay>	-
float	int32	float	-
3DWM : 55 3DCM : 120	1	0	-
Frame periodicity in ms. Refer to the note below[2].	Trigger select. Always set to 1. Only the software trigger is supported.	Frame trigger delay in ms.	-

[1] **numLoops:** As illustrated in Figure 7, this parameter is the total number of chirps transmitted per TX within a single frame. Along with the chirp timing parameters in Section 3.1.2, this parameter configures the motion sensitivity (i.e., velocity resolution) of the detection layer and the maximum detection range based on SNR (please refer to Section 4 for details). Due to the smaller radial velocities observed in overhead scenarios, the velocity resolution of the 3D ceil-mount processing chain is configured as higher than the 3D wall-mount counterpart. This parameter affects the radar cube size in Figure 7 and the required processing time [1], which are carefully optimized for both wall-mount and ceil-mount chains.

[2] **framePeriodicity:** This parameter configures the frame rate of the processing chain. The given values are selected according to the total chirping and the required processing times [1] for each demo. The user must always ensure that the total chirping time (if modified) and the required processing time can fit into a single frame period.

3.2. Detection Layer Parameters

In the following subsections, the detection layer configuration parameters summarized in Table 1 and Table 2 are detailed. The parameters listed in the subsequent tables are in **row-major order**. Each table presents the data type and description of each parameter along with the suggested values in **3D Wall-Mount (3DWM)** and **3D Ceil-Mount (3DCM)** scenarios (please refer to Appendix 6.1 for the template of the tables).

3.2.1. Dynamic Scene Range-Angle Heatmap Configuration

dynamicRangeAngleCfg			
<subFrameIdx>	<angleSearchStep>	<rangeAngleDiagonalLoading>	<rangeAngleEstMethod>
int8	float	float	uint8
-1	3DWM : 0.75 3DCM : 7	3DWM : 0.001 3DCM : 0.001	3DWM : 1 3DCM : 2
Subframe index. Always set to -1.	Angle search granularity in degrees. 0.75 deg is the azimuth search step for the 3D wall-mount scenario. 7 deg is both coarse-grained azimuth-elevation search steps for the 3D ceil-mount scenario.	Range-angle domain diagonal loading factor when constructing the spatial covariance matrix.	1: Range azimuth detection and then elevation estimation. The suggested method for the 3D wall-mount scenario. 2: Range-elevation-azimuth detection, with angle domain zoom-in. The suggested method for the 3D ceil-mount scenario.
<dopplerEstMethod>	-	-	-
uint8	-	-	-
0	-	-	-
0: Single peak search in the Doppler domain. The only method supported.	-	-	-

For the dynamic scenes, the range-angle heatmap estimation step uses Capon beamforming in the 2D range-azimuth domain for the 3D wall-mount chain and 3D range-azimuth-elevation domain for the 3D ceil-mount chain. The configuration parameters are detailed below.

– **subFrameIdx = -1** This parameter is introduced to follow the SDK software architecture [5], although we do not support more than one subframe. Therefore, this field is set to -1 for all the corresponding commands in the 3D people tracking chain.

- **angleSearchStep:** This parameter defines the search granularity (i.e., the inter-bin resolution) in the angle domain. For the 3D wall-mount scenarios, this parameter defines only the azimuth inter-bin resolution because the elevation inter-bin resolution is defined in dynamic2DAngleCfg discussed in Section 3.2.3. For the 3D ceil-mount scenarios, this parameter defines both the azimuth and elevation inter-bin resolutions when creating the coarse azimuth-elevation heatmaps, which will be zoomed-in at finer angles via dynamic2DAngleCfg command, as discussed in Section 3.2.3.

  This field should not be set to a value smaller than 0.75 for the 3D wall-mount and 7 for the 3D ceil-mount scenarios due to memory limitation. Besides, because this parameter defines the angle inter-bin resolution, the user should also adjust reference and guard window sizes of the CFAR configuration (i.e., refWinSizeAngle and guardWinSizeAngle in dynamicRACfarCfg discussed in Section 3.2.2) when changing this parameter. Finally, this parameter is also antenna pattern dependent. Based on the FOV of the radar sensor dictated by the antenna pattern, which is configured by fovCfg in Section 3.3.4, the angle search resolution should be adjusted accordingly to limit the required memory and computation power by the Capon beamformer. Please refer to the implementation guide in [1] for the benchmarks in MIPS and memory. The details about the steering vector design for the Capon beamformer are also discussed in [1].

- **rangeAngleDiagonalLoading:** This is the diagonal loading parameter for constructing a spatial covariance matrix for Capon beamforming in range-angle heatmap estimation. This diagonal loading factor is added to the covariance matrices to ensure a stable inversion. 0.001 is the best empirical setting we obtained from testing for both 3D wall-mount and ceil-mount scenarios. We would strongly suggest not changing the setting. This parameter affects the angle spectrum calculation and the Doppler estimation for each range bin. Hence, changing this parameter could cause significant angle and Doppler estimation errors. In general, the larger the diagonal loading, the stable the inverse of Rn matrix, but worse the SNR. Hence, we set the diagonal loading to be smaller to have a better SNR in the range-azimuth detection.

- **rangeAngleEstMethod:** This is the critical parameter that defines the signal processing chain. For the 3D wall-mount application using the IWR6843 ISK antenna pattern (see Section 3.3.1), this parameter is set to 1. In this method, the range-azimuth heatmap is first generated. After the CFAR step (see Section 3.2.2), for each detected [range, azimuth] pair, a 1D elevation heatmap is calculated for the elevation estimation (see Section 3.2.3). For IWR6843 ODS/AOP type of antenna patterns (see Section 3.3.1) in 3D ceil-mount applications, this parameter is set to 2. This method uses a full range-azimuth-elevation heatmap for range and coarse azimuth/elevation estimation. After the CFAR detection (see Section 3.2.2), a 2D zoom-in is performed for the detected azimuth-elevation bin to find the finer angle estimations (see Section 3.2.3).

- **dopplerEstMethod:** The current implementation only supports a single peak search in the Doppler domain. In this mode, the strongest (i.e., maximum) peak in the Doppler spectrum is selected as the radial velocity estimation of the detected points.

3.2.2. Dynamic Scene CFAR Configuration

dynamicRACfarCfg			
<subFrameIdx>	<cfarDiscardLeftRange>	<cfarDiscardRightRange>	<cfarDiscardLeftAngle>
int8	uint8	uint8	uint8
-1	3DWM : 4 3DCM : 10	3DWM : 4 3DCM : 1	3DWM : 2 3DCM : 1
Subframe index. Always set to -1.	Samples discarded on the left (i.e., closest points to the sensor) of the range domain.	Samples discarded on the right (i.e., furthest points from the sensor) of the range domain.	Samples discarded on the left of the angle domain.
<cfarDiscardRightAngle>	<refWinSizeRange>	<refWinSizeAngle>	<guardWinSizeRange>
uint8	uint8	uint8	uint8
3DWM : 2 3DCM : 1	3DWM : 8 3DCM : 8	3DWM : 12 3DCM : 8	3DWM : 4 3DCM : 6
Samples discarded on the right of the angle domain.	Reference window size (in samples) of range-CFAR.	Reference window size (in samples) of angle-CFAR.	Guard window size (in samples) of range-CFAR.
<guardWinSizeAngle>	<rangeThre>	<angleThre>	<sidelobeThre>
uint8	float	float	float
3DWM : 8 3DCM : 4	3DWM : 5 3DCM : 4	3DWM : 8 3DCM : 6	3DWM : 0.4 3DCM : 0.5
Guard window size (in samples) of angle-CFAR.	Threshold (i.e., scaling) factor of range-CFAR in the linear scale.	Threshold (i.e., scaling) factor of angle-CFAR in the linear scale.	Sidelobe threshold in the linear scale to declare a local peak as a valid detection (i.e., to confirm it is not a sidelobe of the maximum peak).
<enable2ndPass>	<dynamicFlag>	-	-
uint8	uint8	-	-
1	3	-	-
0: Second-pass CFAR disabled. 1: Second-pass CFAR enabled. The suggested method in dynamic scene CFAR.	0 or 1: dynamic CFAR disabled 2: Range dynamic CFAR enabled 3: Range and Angle dynamic CFAR enabled	-	-

For the 3D wall-mount chain, the 2-pass CFAR algorithm is applied directly to the generated 2D range-azimuth heatmap. For the 3D ceil-mount chain, the generated 3D range-azimuth-elevation heatmap is first reshaped into the 2D range-angle heatmap with one dimension on the range and another dimension on the angle (see Section 2.2.4). The same 2-pass CFAR algorithm is then applied to the reshaped 2D range-angle heatmap.

The detection algorithm is a 2-pass CFAR with a first-pass scan on each angle that checks the CFAR condition across the range bins. For each detected point at (rangeIndex, angleIndex), the second CFAR pass (when enabled by enable2ndPass) checks the condition across the angle bins to confirm the first pass detections. For both passes, the CFAR CASO (cell average small of) method is used, i.e., the smallest average of the left and right reference windows is used as noise estimation for the CFAR detection (see Section 4.2.3 for details).

The following search boundary parameters determine the edges of the CFAR search domain in both range and angle dimensions.

- **cfarDiscardLeftRange:** This is the number of samples on the beginning side that will not be included in the first-pass CFAR search (in the range domain). Depending on the chirp configuration and derived inter-bin range resolution, this setting dictates the range near the sensor that will be excluded from the detection.

  For the example chirp configuration of the 3D wall-mount chain, a value of 4 in this field implies that the 34cm (4x8.5cm, where 8.5cm is the range bin resolution) within the sensor will not have any detection from the CFAR. Similarly, in the example chirp configuration of the 3D ceil-mount chain, a value of 10 in this field implies that the 85cm (10x8.5cm, where 8.5cm is the range bin resolution) within the sensor will not have any detection from the CFAR.

  Discarding these range bins also reduces the memory requirement and processing time. If the end application is tight in memory or processing time and there is no interest in detecting the close-by range, these bins can be discarded to save resources.

- **cfarDiscardRightRange:** This is the number of samples on the ending side that will not be included in the first-pass of the CFAR search (in the range domain). Depending on the chirp configuration and derived inter-bin range resolution, this setting dictates the range furthest away from the sensor (typically the range boundary of the scene) that will be excluded from the detection.

  For the example chirp configuration of the 3D wall-mount chain, a value of 4 in this field implies the furthest 34cm from the sensor will not have any detection from the CFAR. Similarly, in the example chirp configuration of the 3D ceil-mount chain, a value of 1 in this field implies that the furthest 8.5cm from the sensor will not have any detection from the CFAR.

  Discarding these range bins also reduces the memory requirement and processing time. If the end application is tight in memory or processing time and there is no interest in detecting the far range, these bins can be discarded to save resources.

- **cfarDiscardLeftAngle and cfarDiscardRightAngle:** These two parameters are similar to cfarDiscardLeftRange and cfarDiscardRightRange, but rather in the angle domain, where the second pass of the CFAR is applied.

The following CFAR window parameters in both range and angle dimensions are used to define a sliding window to calculate the local noise floor to be compared with the cell under test (CUT) (see Section 4.2.3 for details).

- **refWinSizeRange and refWinSizeAngle:** These fields are the CFAR reference window sizes (in terms of the number of samples) for the first (range domain) and second (angle domain) passes. The values of [8, 12] for the 3D wall-mount and [8, 8] for the 3D ceil-mount are empirically the best choices, and the user may want to use it as a default setting.

- **guardWinSizeRange:** This field is the CFAR guard window size (in terms of the number of samples) in the range domain. When calculating the detection threshold for the CUT, the left and right guardWinSizeRange of samples will be excluded from noise accumulation. If we have a target with an area of 0.5m^2 reflecting radar energy, we do not want samples in those areas being counted as noise samples to raise the detection threshold. This way, there will be a richer point cloud detection out of the CFAR step.

  For the example chirp configuration of the 3D wall-mount chain, a value of 4 in this field implies the 34cm on each side of the CFAR CUT that should not be counted into noise power for CFAR detection. Similarly, in the example chirp configuration of the 3D ceil-mount chain, a value of 6 in this field implies the 51cm on each side of the CFAR CUT that should not be counted into noise power for CFAR detection.

  The user should adjust the setting based on the chirp configuration and derived inter-bin resolution, as well as the typical target size within the scene. These settings have already been adjusted for the desired effect on people as intended targets with the provided chirp configuration.

- **guardWinSizeAngle:** Similar to guardWinSizeRange, this field is the CFAR guard window size (in terms of the number of samples) in the angle domain. Again, this has been empirically adjusted for the desired effect on people as the intended targets of interest with the provided chirp configuration.

The following range-angle CFAR threshold parameters are defined as ratios of the local noise floor to the CUT, above which a detected point is declared. The lower the thresholds, the more false detections will occur on random noise, object sidelobes, and other spurious environmental effects. The higher the thresholds, the fewer points from the objects of interest are detected.

- **rangeThre:** This is the relative threshold for the first-pass (in the range domain). After noise power (noisePower) is calculated based on the refWinSizeRange and guardWinSizeRange, the final detection threshold will be calculated by (noisePower * rangeThre), where noisePower is the estimated noise power in the reference windows. If the power of CUT is greater than the threshold, it will be declared as a detected point.

  The threshold in range domain settings impacts detection performance significantly. The lower the threshold, the more low confident detections may show up. The user would see more noisy points around the target, more multipath ghost targets forming between the real target and strong reflectors such as metal beams and walls, and more angle and Doppler errors from the low confident point. On the other hand, if the threshold is too high, the user will lose a detected point that potentially carries valuable information about the target's angle/size and Doppler. Loss of this information would affect the performance of the following modules, such as tracker and classifier, which rely heavily on the statistics/distribution of the range/angle/Doppler information of detected points from a target. The suggested values are the best empirical numbers TI obtained, but the user might need to adjust them for their customized setup.

If <dynamicFlag> is enabled with either a value of “2” or “3” the rangeThre will be multiplied by a function which simulates the magnitude of the radar signal over range. This equation is calculated in RADARDEMO_detectionCFAR.c in the function RADARDEMO_detectionCFAR_dynamic().

- **angleThre:** Similar to rangeThre, this is the relative threshold for the second pass (in the angle domain). Similarly, the setting of this threshold also affects the overall performance of the system. If set too low, the user will see an angle spread from the target (the rainbow effect around the target), which may mislead the tracker. If set too high, we again lose valuable information. The suggested values are the best empirical numbers TI obtained, but the user might need to adjust them for their customized setup.

  If <dynamicFlag> is enabled with either a value of “3” the angleThre will be multiplied by a function which simulates the magnitude of the radar signal over angle. This equation is calculated in RADARDEMO_detectionCFAR.c in the function RADARDEMO_detectionCFAR_dynamic().

The following two parameters configure the second-pass search step in the angle domain to confirm the range-domain detected points.

- **sidelobeThre:** This parameter is used in combining with the second-pass search. If the second-pass is enabled (i.e., enable2ndPass = 1), the algorithm will try to confirm the range-domain detected points along the angle domain using the CFAR-CASO with a threshold factor configured by the angleThre parameter discussed above.

  If the range-point is not confirmed by the second-pass angle-CFAR (i.e., enable2ndPass = 0), the angle-CFAR search will be skipped, and the algorithm will check whether the detected range-point is a local maximum in the 2D range-angle domain or not. If the detected range-point is a local maximum in the angle domain and if its power exceeds the (sidelobeThre * peakPower), where peakPower is the power of the strongest peak in the same range bin, the algorithm will confirm it as a detected range-azimuth point.

- **enable2ndPass:** This parameter indicates the second-pass CFAR confirmation is enabled or not, where the effect of both approaches (i.e., when the second-pass is enabled or disabled) is detailed above. This field is set to 1 (i.e., the second-pass CFAR is enabled) by default in both 3D wall-mount and ceil-mount scenarios when detecting the dynamic scenes.

- **dynamicFlag:** This parameter indicates if a dynamic CFAR threshold is used in just the range direction (i.e. dynamicFlag = 2), is used in the range and angle directions (i.e. dynamicFlag = 3), or is not used at all (i.e. dynamicFlag = 0 or 1). The dynamic CFAR threshold is a threshold which varies based on the strength of the signal expected. The expected signal strength varies relative to range and relative to angle.

3.2.3. Dynamic Scene 2D Angle Estimation Configuration

The field definitions of dynamic2DAngleCfg are different for the range-angle heatmap estimation methods (i.e., rangeAngleEstMethod) 1 (i.e., the suggested method for the 3D wall-mount scenario) and 2 (i.e., the suggested method for the 3D ceil-mount scenario) configured by dynamicRangeAngleCfg in Section 3.2.1. Hence, in this section, we will discuss the definitions of each version in separate tables.

dynamic2DAngleCfg (rangeAngleEstMethod = 1, which is suggested for 3DWM)			
<subFrameIdx>	<elevSearchStep>	<angleDiagonalLoading>	<maxNpeak2Search>
int8	float	float	uint8
-1	3	0.03	1
Subframe index. Always set to -1.	Elevation search granularity in degrees.	Angle domain diagonal loading factor when constructing the spatial covariance matrix.	The number of peaks to search in elevation. Single peak search is supported only (i.e., always set to 1).
<peakExpSamples>	<elevOnly>	<sidelobeThre>	<peakExpRelThre>
uint8	uint8	float	float
0	1	0.3	0.85
Number of samples on each side to expand the peak of the elevation heatmap. Peak expansion is not supported in this method (i.e., always set to 0).	1: Elevation estimation only. The only method supported.	Sidelobe threshold in the linear scale to declare a local peak as the detected point. This field is reserved for future versions when the multi-peak search is enabled.	Relative threshold (to the peak power) in the linear scale to include the neighbors of the peak as detection. This field is reserved for future versions when the peak expansion is enabled.
<peakExpSNRThre>	-	-	-
float	-	-	-
8	-	-	-
Linear SNR threshold for the peak to enable peak expansion in the heatmap. This field is reserved for future versions when the peak expansion is enabled.	-	-	-

When the range-angle heatmap estimation method 1 is used (i.e., rangeAngleEstMethod = 1), for each point detected in the range-azimuth domain, a Capon beamforming algorithm is applied to generate the elevation spectrum, which will be used to estimate the elevation angle of the detected point according to the configuration parameters detailed below:

- **elevSearchStep:** This parameter defines the search resolution (inter-bin resolution). Due to memory limitation, it should not be set to a value smaller than 1. This parameter is also the antenna pattern dependent. Based on the FOV of the radar sensor dictated by the antenna pattern, which is configured by fovCfg in Section 3.3.4, the elevation search step should be adjusted accordingly to limit the required memory and computation power by the Capon beamformer. Please refer to the implementation guide in [1] for the benchmarks in MIPS and memory. The details about the steering vector design for the Capon beamformer are also discussed in [1].

- **angleDiagonalLoading:** This is the diagonal loading parameter for constructing a spatial covariance matrix for Capon beamforming in azimuth-elevation heatmap estimation. 0.03 is the best empirical setting we obtained from testing. We would strongly suggest not changing the setting. This parameter affects the spatial covariance matrix estimation. Therefore, it affects the angle spectrum calculation and the Doppler estimation for each range bin. Hence, changing this parameter could cause significant angle and Doppler estimation errors.

- **maxNpeak2Search:** Current implementation only supports a single peak search in the elevation domain. Using the ISK antenna pattern (see Section 3.3) in wall-mount scenarios, there is basically no need for a multi-peak search due to the less virtual antennas in the elevation domain (see Section 3.3 for the antenna patterns). This hook enables future support of multi-peak search to enrich the point cloud. The possible future support of a multi-peak search feature will be used with the following parameter to declare multiple local peaks as the detected point:

    - **sidelobeThre:** The relative threshold (comparing to the global maxima) to declare a local peak as the detected point. This parameter is reserved for future versions when the multi-peak search feature is available.

- **peakExpSamples:** Current implementation does not support peak expansion around the detected elevation peak. Using the ISK antenna pattern (see Section 3.3) in wall-mount scenarios, there is basically no need for peak expansion due to the less virtual antenna in the elevation domain. This hook enables future support of peak-expansion to enrich the point cloud. The possible future support of a peak-expansion feature will be used with the following two parameters to include the configured number of neighboring samples of the peak as detection:

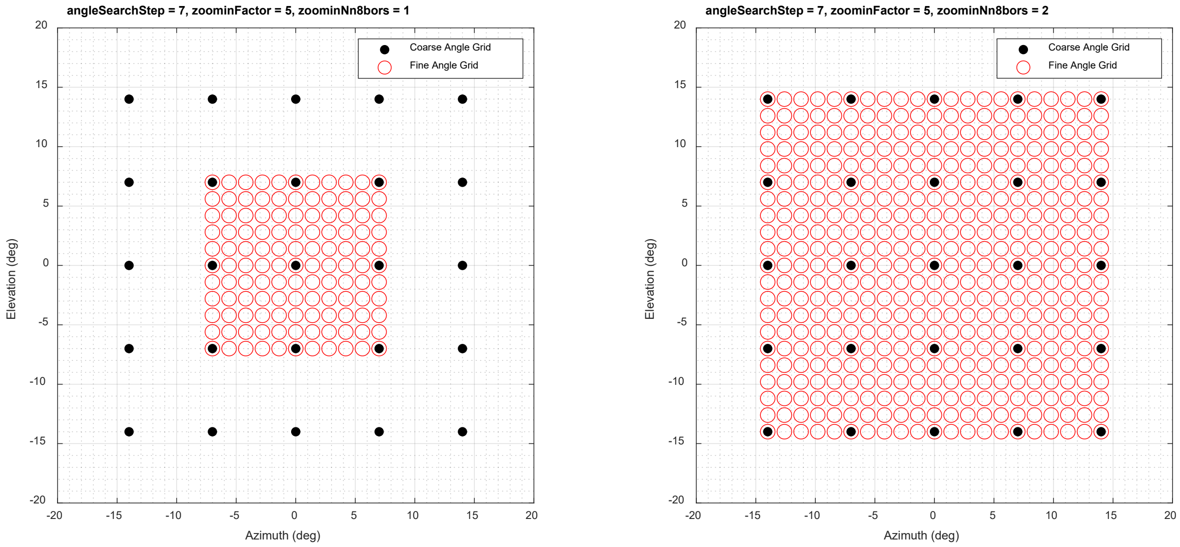
    - **peakExpRelThre:** The relative threshold to include the neighbor of the peak as the detected point. This parameter is reserved for future versions when the peak-expansion feature is available.

    - **peakExpSNRThre:** The SNR threshold for the detected peak to allow peak expansion around it. This parameter is reserved for future versions when the peak-expansion feature is available.

- **elevOnly:** In the current implementation, we only perform elevation estimation per detected azimuth bin.

dynamic2DAngleCfg (rangeAngleEstMethod = 2, which is suggested for 3DCM)			
<subFrameIdx>	<zoominFactor>	<zoominNn8bors>	<peakExpSamples>
int8	uint8	uint8	uint8
-1	5	1	1
Subframe index. Always set to -1.	Zoom-in factor in both elevation and azimuth domains.	Number of coarse neighboring angle bins of zoom-in. Only 1 is supported.	Number of samples on each side to expand the peak of the zoomed-in azimuth-elevation heatmap.
<peakExpRelThre>	<peakExpSNRThre>	<localMaxCheck>	-
float	float	uint8	-
1	15	2	-
The starting relative threshold (to the peak power) in the linear scale to include the peak's neighbor as detection. This parameter is recalculated in the implementation based on the sharpness of the peak.	Linear SNR threshold for the peak to enable peak expansion in the zoomed-in heatmap.	0: No local maximum check. 1: If the coarse peak is not local maximum in the elevation domain, exclude it from the detection. 2: If the coarse peak is not local maximum in both elevation and azimuth domains, exclude it from the detection.	-

When the range-angle heatmap estimation method 2 is used (i.e., rangeAngleEstMethod = 2), a Capon beamforming algorithm is applied to generate the zoomed-in azimuth-elevation angle spectrums around the detected coarse azimuth-elevation points from the previous 2-pass CFAR step as depicted in Figure 8. In this example, two different zoom-in grids around the detected peak are created with different parameters to estimate the finer azimuth and elevation angles.



(a) **(b)**

Figure 8. An example zoom-in grid to estimate the finer azimuth-elevation angle when the coarse angle search step is 7 deg (angleSearchStep = 7) and : (a) zoominFactor = 5 and zoominNn8bors = 1, (b) for zoominFactor = 5 and zoominNn8bors = 2.

The strongest peak of each zoomed-in spectrum and its neighbors that meet some specific criteria are then selected as the finer azimuth and elevation angles of the detected points according to the configuration parameters detailed below:

– **zoominFactor:** This parameter is used to define the finer angle search (i.e., inter-bin) resolution. Around each coarse-grained azimuth-elevation detection point, a fine-grained azimuth-elevation angular power spectrum is generated with a zoom-in grid, as illustrated in Figure 8. The finer angle search resolution is then computed by angleSearchStep/zoominFactor, where angleSearchStep is the coarse angle search step configured by dynamicRangeAngleCfg (see Section 3.2.1). Due to memory limitation and MIPS, it should not be set to a value bigger than 6. This parameter is also the antenna pattern dependent. Based on the FOV of the radar sensor dictated by the antenna pattern, which is configured by fovCfg in Section 3.3.4, the finer angle grid should be adjusted accordingly to limit the required MIPS by the total number of zoom-in Capon beamformers called.

– **zoominNn8bors:** This is the number of neighboring samples to zoom-in on the coarse angular granularity. As depicted in Figure 8, along with the zoominFactor configured above, this parameter defines the total zoom-in grid size of the fine-grained azimuth-elevation angular power spectrum. For the ODS antenna pattern, 1 (as illustrated in Figure 8a) is the optimal and the only supported value. Since we only perform a single peak search in the zoomed-in azimuth-elevation heatmap, if the zoom-in grid size is bigger than the antenna pattern's angular resolution, we could potentially lose detection.

– **peakExpSamples:** This is the number of neighboring samples on each side of the peaks in the 2D finer angular domain that can also be declared as detected points. The neighboring samples must meet the relative threshold configured in the following peakExpRelThre parameter to expand a peak in the zoomed-in azimuth-elevation heatmap. Besides, the peak must meet the SNR threshold defined in the following peakExpSNRThre parameter to be expanded. The implementation details of the peak expansion are discussed in [1].

- **peakExpRelThre:** This is the starting relative threshold for the peak expansion approach. Inside the implementation, a simple sharpness factor (i.e., acutance) is calculated by (maxPower - minPower)/(maxPower + minPower), where maxPower and minPower are the maximum and minimum powers in the fine-grained azimuth-elevation spectrum, respectively. Then (peakExpRelThre-sharpness) * peakPower will be used as the threshold to include the neighboring points as detection, where peakPower is the power of the peak to be expanded. It is suggested to set the starting relative threshold value to 1, allowing the algorithm to recalculate it based on the sharpness factor. The bigger the sharpness factor means the sharper and better quality of the peak and the smaller threshold in the peak expansion, resulting in more detected points in the generated point cloud.

- **peakExpSNRThre:** This is the SNR threshold to allow peak expansion for the detected peak. Only the detected points with enough SNR that exceeds this threshold will be allowed to perform peak expansion. This field is compared with the coarse peak SNR calculated by the CFAR detection configured by dynamicRACfarCfg in Section 3.2.2.

- **localMaxCheck:** The flag to indicate what local maximum to check to include the detected points for zoom-in and finer angle estimations:

  - **0:** All the points detected in the coarse-grained azimuth-elevation spectrum are allowed to perform zoom-in. In other words, there is no local maximum check with this mode, and even the angle sidelobes of the coarse 2D angle detection are allowed to be zoomed-in.

  - **1:** When this mode is selected, the points detected in the coarse-grained azimuth-elevation spectrum are allowed to perform zoom-in if they are also a local maximum in the elevation domain. In other words, if the coarse peak is not a local maximum in the elevation domain, it will be excluded from the detection list.

  - **2:** When this mode is selected, the points detected in the coarse-grained azimuth-elevation spectrum are allowed to perform zoom-in if they are also a local maximum in both the azimuth and elevation domains. In other words, if the coarse peak is not a local maximum in both elevation and azimuth domains, it will be excluded from the detection list. This mode, along with the zoominFactor, will ensure that only one peak will be zoomed into a finer angle heatmap. Therefore, a single peak search with peak expansion will produce a good non-redundant quality of detected points around the peak.

3.2.4. Static Scene Range-Angle Heatmap Configuration

staticRangeAngleCfg			
<subFrameIdx>	<staticProcEnabled>	<staticAzimStepDeciFactor>	<staticElevStepDeciFactor>
int8	uint8	uint8	uint8
-1	0	3DWM : 8 3DCM : 1	3DWM : 2 3DCM : 1
Subframe index. Always set to -1.	0: Static scene processing disabled. 1: Static scene processing enabled.	Azimuth domain decimation factor.	Elevation domain decimation factor.

The range-angle heatmap estimation step uses Bartlett beamforming in the range-azimuth-elevation domain for the static scenes. The configuration parameters are detailed below:

- **staticProcEnabled:** The static scene processing is enabled by setting this flag to 1. If there is no need for static scene processing, the user can disable it to save MIPS. It is important to note that the user needs to adjust the following angular search parameters for the static scene and the dynamic scene counterparts to keep the processing load within acceptable limits [1]. When the static scene processing mode is disabled, the following parameters have no impact on the processing chain, and they will be ignored.

- **staticAzimStepDeciFactor** and **staticElevStepDeciFactor:** The static scene processing uses lower or equal angular search resolution than dynamic scene processing. Since several buffers and parameters are reused, we only specify the decimation factor here.

  If enabled in the 3D wall-mount chain (i.e., the dynamic range-angle heatmap estimation method rangeAngleEstMethod is 1), with the default setting in this document,

  - staticAzimAngleStep = (staticAzimStepDeciFactor*azimSearchStep) = (8*0.75) = 6 degree
  - staticElevAngleStep = (staticElevStepDeciFactor*elevSearchStep) = (2*3) = 6 degree.

  Note that azimSearchStep and elevSearchStep are defined in commands dynamicRangeAngleCfg (in Section 3.2.1) and dynamic2DAngleCfg (in Section 3.2.3). If enabled in the 3D ceil-mount chain (i.e., the dynamic range-angle heatmap estimation method is 2), both fields should be set to 1 to reuse the same steering vector computed to generate the coarse azimuth-elevation heatmap based on angleSearchStep in dynamicRangeAngleCfg (in Section 3.2.1).

3.2.5. Static Scene CFAR Configuration

staticRACfarCfg			
<subFrameIdx>	<cfarDiscardLeftRange>	<cfarDiscardRightRange>	<cfarDiscardLeftAngle>
int8	uint8	uint8	uint8
-1	3DWM : 6 3DCM : 10	3DWM : 2 3DCM : 1	3DWM : 2 3DCM : 1
Subframe index. Always set to -1.	Samples discarded on the left (i.e., closest points to the sensor) of the range domain.	Samples discarded on the right (i.e., furthest points from the sensor) of the range domain.	Samples discarded on the left of the angle domain.
<cfarDiscardRightAngle>	<refWinSizeRange>	<refWinSizeAngle>	<guardWinSizeRange>
uint8	uint8	uint8	uint8
3DWM : 2 3DCM : 1	3DWM : 8 3DCM : 8	3DWM : 8 3DCM : 8	3DWM : 6 3DCM : 6
Samples discarded on the right of the angle domain.	Reference window size (in samples) of range-CFAR.	Reference window size (in samples) of angle-CFAR.	Guard window size (in samples) of range-CFAR.
<guardWinSizeAngle>	<rangeThre>	<angleThre>	<sidelobeThre>
uint8	float	float	float
3DWM : 4 3DCM : 4	3DWM : 8 3DCM : 8	3DWM : 15 3DCM : 15	3DWM : 0.3 3DCM : 0.3
Guard window size (in samples) of angle-CFAR.	Threshold (i.e., scaling) factor of range-CFAR in the linear scale.	Threshold (i.e., scaling) factor of angle-CFAR in the linear scale.	Sidelobe threshold in the linear scale to declare a local peak as a valid detection (i.e., confirm it is not a sidelobe of the maximum peak).
<enable2ndPass>	<dynamicFlag>	-	-
uint8	N/A	-	-
0	0	-	-
0: Second-pass CFAR disabled. The suggested method in static scene CFAR. 1: Second-pass CFAR enabled.	Always set to 0. This flag has no impact on the chain and is not used anymore.	-	-

Similar to the detection approach applied to the dynamic range-azimuth heatmap, a 2-pass CFAR algorithm is applied to the 3D range-azimuth-elevation heatmap created from the static scene. Before the CFAR detection, the generated 3D range-azimuth-elevation heatmap is first reshaped into a 2D range-angle heatmap with one dimension on the range and another dimension on the angle (due to the same reason discussed for the dynamic detection step). In the reshaped version, azimuth-elevation angle is compressed into one dimension as (aziInd, eleInd) → (aziInd + eleInd * azimDim).

The first-pass CFAR is applied on each angle of the 2D range-angle heatmap to check the CFAR condition across range bins. In the second pass, each detected peak in the range domain (rangeIndex) is confirmed across the angle bins according to the second-pass configuration parameters (sidelobeThre, enable2ndPass) defined in Section 3.2.5.

Static scene CFAR configuration has the same fields (and descriptions) as the dynamic scene CFAR configuration defined in Section 3.2.2. The reference and guard window sizes (refWinSizeAngle and guardWinSizeAngle) are adjusted due to the coarser angle step. A higher threshold is suggested to pick up the strong reflections only.

After the detection, while the peak information in the range domain is reported directly, a simple interpolation step is applied to the detected azimuth and elevation peaks using their neighbors. This step reports a finer azimuth-elevation estimate in the point cloud. Hence, no further zoom-in algorithm is applied for finer angle accuracy.

3.2.6. Fine Motion Detection Mode Configuration

In the 3D people tracking demo, a processing mode is added to detect and track static people with fine motions (standing, sitting still, sleeping, etc.). In this mode, the processing chain creates two different chirp blocks (see Section 3.1.3 for radar cube definitions) in the memory and runs the detection layer processing chain (see Section 2) on both chirp blocks in different time slots. The algorithm flows through the major steps depicted in the block diagram in Figure 9.

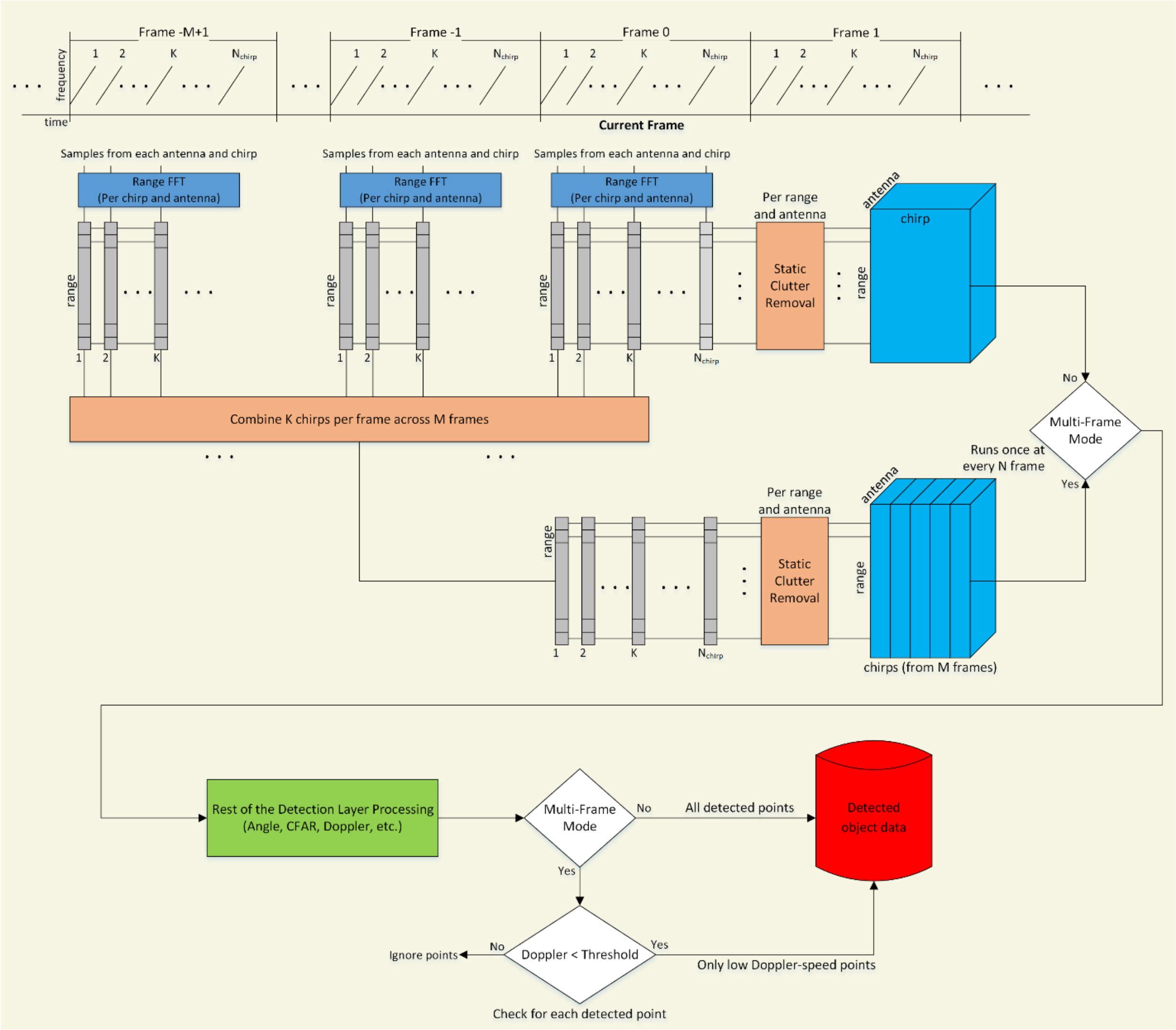


Figure 9. The signal processing chain of the fine motion mode.

As detailed in Figure 9, the first radar cube is created using all the available chirps of the current frame, and the second radar cube is created using the portion of the current frame and previous frames. In this processing mode, the radar cube from the single frame is utilized to detect dynamic people in the scene. The second radar cube that combines the chirp sub-blocks from the current and previous frames achieves a longer chirping window when detecting a static person in the same scene. The tracker layer

[2] then performs the logic to properly use each mode's detection layer output according to the estimated velocity of any tracked object. In this logic, the tracker utilizes the detection layer output of the single-frame mode when a track is decided as dynamic. Similarly, when the track becomes static, the detection layer output of the multi-frame mode is utilized.

When creating the multi-frame block, the number of frames to be combined and the number of chirps selected from each frame are software configurable and depend on the frame rate, the number of chirps available in a single frame, and the velocity resolution required to detect fine motions. Hence, the total observation time window is configured based on the desired velocity resolution when configuring these parameters. The processing chain then decides how many frames to be combined and the number of chirps selected from each frame.

When creating the fine-motion mode radar cube using chirps across multiple frames, the first K chirps per frame is utilized (instead of any subsampling) to keep the maximum unambiguous velocity the same as the single-frame block. Besides, the processing chain creates the fine-motion mode radar cube the same size as the single-frame block to fit the rest of the processing chain.

In this scheme, the main processing blocks (angle, Doppler, and detection) are then run on two different radar cubes (from the single or multiple frames) in time-division mode. The processing chain runs on the multi-frame block once at every N frame (software configurable), and the conventional single-frame block will be utilized in the remaining N-1 time slots. Therefore, at every frame, the processing chain creates and outputs the point cloud (including range, angle, Doppler, and SNR) either from single-frame (N-1 times at every N frame) or multi-frame blocks (once at every N frame).

Finally, when the processing chain runs on the multiple-frames radar cube, the points generated from dynamic tracks are filtered out to improve the tracker's robustness. Hence, in the multi-frame processing mode, only the detected points from static targets are desired to be generated to properly run a robust tracker layer logic. The Doppler threshold (in terms of the number of one-sided Doppler bins) is software configurable, as discussed in the following paragraphs.

fineMotionCfg			
<subFrameIdx>	<fineMotionProcEnabled>	<fineMotionObservationTime>	<fineMotionProcCycle>
int8	uint8	float	uint16
-1	1	3DWM : 1.0 3DCM : 2.0	10
Subframe index. Always set to -1.	0: Fine motion processing disabled. 1: Fine motion processing enabled.	Total observation time (in seconds) is utilized to compute the number of frames combined and the number of chirps selected from each frame.	The processing cycle of the fine-motion mode (in frames). The algorithm runs the multi-frame block once at every N frame.

<fineMotionDopplerThrIdx>	-	-	-
uint16	-	-	-
2	-	-	-
The Doppler threshold (in terms of the number of one-sided Doppler bins) is utilized when filtering the dynamic points in the fine-motion mode.	-	-	-

fineMotionProcEnabled: When the fine motion detection mode is enabled using this command, due to the required memory to store the chirp sub-blocks from the current and previous frames, the user should configure the total number of chirps in a single frame (frameCfg->numLoops in Section 3.1.3) as half of the original value. Using the half number of chirps in a single frame, the performance will be affected in two ways:

- The velocity resolution within a single frame will reduce. However, the fine motion mode will recover this performance degradation by processing data across multiple frames.
- The total SNR will reduce. To recover this SNR degradation, the user can enable BPM-MIMO mode across two TX antennas (see Section 3.1.3 and 3.2.7 for details about the supported BPM-MIMO mode) or reduce the applied TX backoff power (see Section 3.1.2 and Section 4.2.2 for details about the transmit power parameters).

Note that this command is not mandatory to run the 3D people tracking demo. If the user does not add this command to the configuration file (see Table 1 and Table 2), the demo disables the fine motion detection mode by default. To achieve the best performance when detecting very fine motions in the scene, it is recommended to enable this mode. When the fine motion mode is disabled, the following parameters have no impact on the processing chain, and they will be ignored.

fineMotionObservationTime: When the fine motion detection mode is enabled, this parameter is utilized to compute the number of frames combined (i.e., the parameter M in Figure 9) and the number of chirps selected from each frame (i.e., the parameter K in Figure 9). When the person is static, the most observable motion signature from the human body is breathing. Hence, configuring the observation time as around 1 or 2 seconds can help to distinguish a person from a fully static clutter. Due to the smaller radial velocities observed in overhead scenarios, the observation window in the 3D ceil-mount processing chain is configured as higher than the 3D wall-mount counterpart.

Based on the frame rate and the number of chirps available in a single frame, the processing chain computes the parameters M and K that create the fine-motion mode radar cube the same size as the single-frame block to fit the rest of the processing chain. For example, when 1 seconds of observation time is configured in the 3D wall-mount demo, three chirps per frame are combined across 16 frames to create a 48 chirps block (the same number of chirps per frame when the fine-motion is enabled). When the frame period is configured as 55ms, this mode will achieve ~880ms observation time.

fineMotionProcCycle: When the fine motion detection mode is enabled, this parameter configures the processing cycle of the multi-frame mode in terms of the number of frames. As depicted in Figure 9, the algorithm runs the multi-frame block once at every N frame. The remaining N-1 slots will be utilized for the conventional single-frame mode.

This parameter should be tuned to make a tradeoff between the dynamic and static people tracking accuracy. The static person detection performance may degrade when the fine-motion processing cycle is configured as too high (i.e., less frequent). On the other hand, when the fine-motion processing cycle is configured as too low (i.e., more frequent), the dynamic person detection performance may degrade. 10 is the best empirical setting we obtained from testing for both 3D wall-mount and ceil-mount scenarios.

fineMotionDopplerThrIdx: When the fine motion detection mode is enabled, this parameter configures the Doppler threshold to filter out the points generated from dynamic tracks. This parameter is defined in terms of the number of one-sided Doppler bins. For example, when 2 is configured in this parameter, the processing chain extracts the points whose Doppler bins fall in the [-2:2] interval and ignores the remaining points. This parameter helps to improve the tracker's robustness and avoid additional artifacts created when dynamic targets are tracked.

When the Doppler threshold is configured as too low, fewer points are extracted from the static person, degrading the performance. On the other hand, when the Doppler threshold is configured as too high, more points are extracted from both dynamic and static tracks, which may increase the false alarms. 2 is the best empirical setting we obtained from testing for both 3D wall-mount and ceil-mount scenarios.

To note that the processing chain tags the extracted low Doppler points from the fine motion mode as fully static (i.e., forces their Doppler to zero). This approach helps the tracker to classify the extracted points as dynamic and static when applying a proper logic to dynamic and static tracks (see [2][3] for details).

3.2.7. BPM Mode Configuration

The following bpmCfg command, along with the previously discussed chirpCfg command (see Section 3.1.3), configure the TX antenna multiplexing scheme (BPM or TDM) to enable the MIMO mode on the sensors. In BPM-MIMO mode, with the simultaneous transmission on both TX antennas, the total transmitted power per chirp interval is increased, which translates to an SNR improvement of 3dB. Hence, in the 3D people tracking demo, it is suggested to use BPM-MIMO mode to improve the detection performance, especially when the fine motion detection mode is enabled using half number of chirps (see Section 3.2.6).

bpmCfg			
<subFrameIdx>	<enabled>	<chirp0Idx>	<chirp1Idx>
int8	uint8	uint8	uint8
-1	1	0	2
Subframe index. Always set to -1.	Enable/disable the BPM mode. Set to 0 to disable (i.e., all TX antennas are in TDM mode) and 1 to enable (i.e., two TX antennas are in BPM mode and 1 TX antenna is in TDM mode).	If BPM is enabled in previous argument, this is the chirp index for the first BPM chirp. It will have phase 0 on both TX antennas (TXA+ , TXB+). Note that the chirpCfg command (in Section 3.1.3) for this chirp index must have both TX antennas enabled. If BPM is disabled, this argument has no impact on the demo.	If BPM is enabled, this is the chirp index for the second BPM chirp. It will have phase 0 on TXA and phase 180 on TXB (TXA+ , TXB-). Note that the chirpCfg command (in Section 3.1.3) for this chirp index must have both TX antennas enabled. If BPM is disabled, this argument has no impact on the demo.

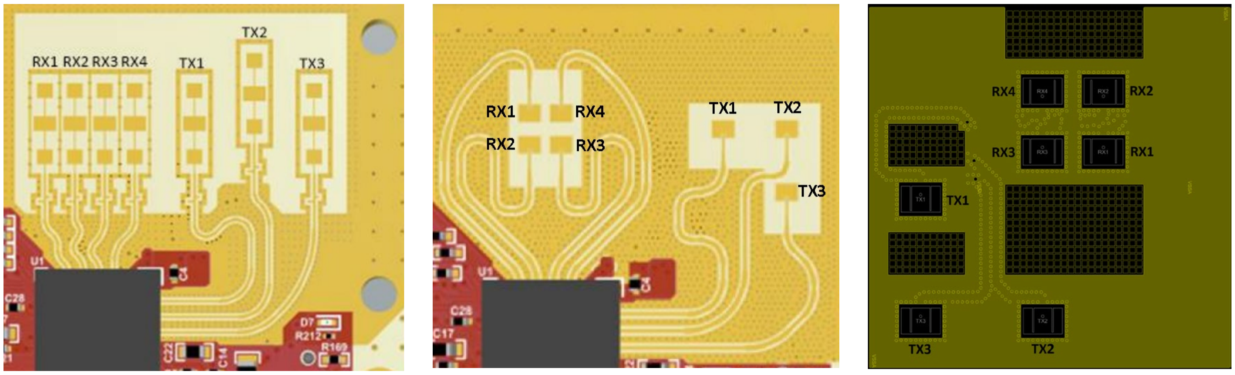
The following table summarizes the supported combinations of the BPM/TDM MIMO scheme and the chirp configurations for each board. As discussed in Section 3.1.3, the 3D people tracking demo supports the BPM-MIMO scheme with only two TX antennas. In this mode, two TX antennas should transmit simultaneously in BPM mode, and the third TX antenna should be in TDM mode. To achieve the best performance in the 3D wall-mount processing chain (where this scheme is tested), the two TX antennas in BPM-MIMO mode should be in the same azimuth axis (for details, please refer to the antenna patterns in Section 3.3).

	ISK	ODS	AOP
BPM-MIMO	chirpCfg 0 0 0 0 0 0 0 5 chirpCfg 1 1 0 0 0 0 0 2 chirpCfg 2 2 0 0 0 0 0 5 bpmCfg -1 1 0 2	chirpCfg 0 0 0 0 0 0 0 3 chirpCfg 1 1 0 0 0 0 0 3 chirpCfg 2 2 0 0 0 0 0 4 bpmCfg -1 1 0 1	chirpCfg 0 0 0 0 0 0 0 1 chirpCfg 1 1 0 0 0 0 0 6 chirpCfg 2 2 0 0 0 0 0 6 bpmCfg -1 1 1 2
TDM-MIMO	chirpCfg 0 0 0 0 0 0 0 1 chirpCfg 1 1 0 0 0 0 0 2 chirpCfg 2 2 0 0 0 0 0 4 bpmCfg -1 0 0 0		

Note that this command is not mandatory to run the 3D people tracking demo. If the user does not add this command to the configuration file (see Table 1 and Table 2), the demo disables the BPM mode by default and configures the TDM-MIMO scheme for each TX antenna if the chirpCfg command (see Section 3.1.3) is configured according to the guidelines.

3.3. Board Related Parameters

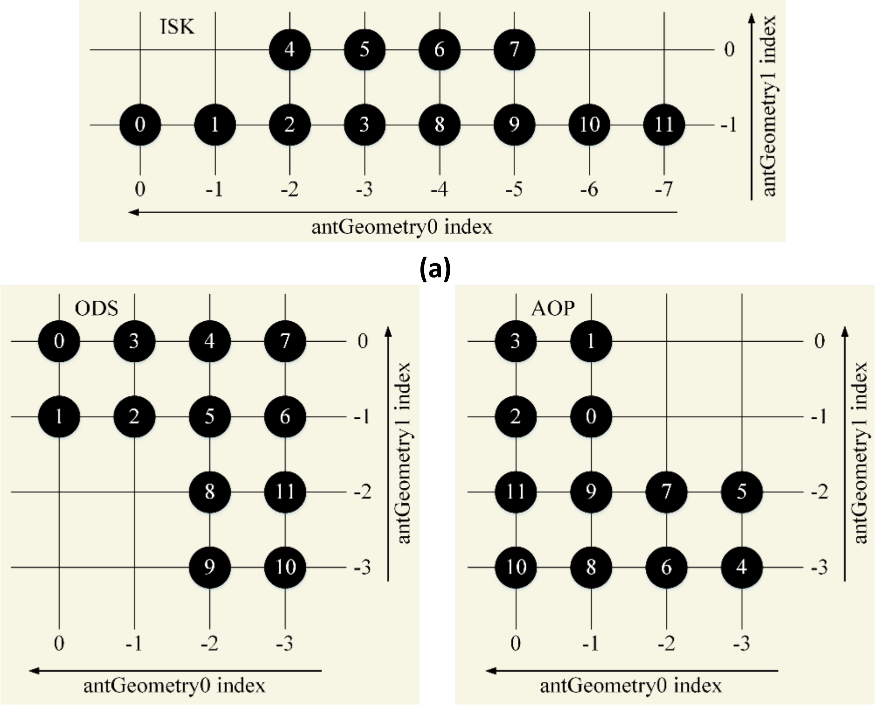
The detection layer running on the IWR6843 mmWave sensors supports both ISK, ODS, and AOP type of antenna patterns shown in Figure 10.



(a) (b) (c)

Figure 10 – Physical antenna patterns for: (a) ISK, (b) ODS, and (c) AOP EVMs

The 3D people tracking demo is developed to be run in a typical far-field scenario, assuming that the distance to the targets (i.e., people) is much larger than the distance between the transmit and receive elements of the IWR6843 MIMO arrays in Figure 10. This assumption leads to a conventional monostatic approximation, where all the transceiver pairs in each MIMO array with three transmit and four receive elements are paired to approximate a monostatic radar operation and create 12 virtual monostatic antennas. Figure 11 illustrates the position index of each virtual element of ISK, ODS, and AOP patterns.



(a)

(b) (c)

Figure 11 – Virtual antenna patterns for: (a) ISK, (b) ODS, and (c) AOP EVMs

In the following sections, we define the configuration parameters for ISK, ODS, and AOP antenna patterns depicted in Figure 10 and Figure 11.

3.3.1. Antenna Geometry Configuration

Antenna geometry parameters antGeometry0 and antGeometry1 define the virtual antennas' physical location index (0, -1, -2, …) in the azimuth and elevation domains, respectively, as illustrated in Figure 11. The following tables summarize these parameters for ISK, ODS, and AOP evaluation boards.

antGeometry0					
<virtAntIdx 0>	<virtAntIdx 1>	<virtAntIdx 2>	<virtAntIdx 3>	<virtAntIdx 4>	<virtAntIdx 5>
int8	int8	int8	int8	int8	int8
ISK : 0 ODS : 0 AOP : -1	ISK : -1 ODS : 0 AOP : -1	ISK : -2 ODS : -1 AOP : 0	ISK : -3 ODS : -1 AOP : 0	ISK : -2 ODS : -2 AOP : -3	ISK : -3 ODS : -2 AOP : -3
Location index of each virtual antenna in the azimuth domain (see Figure 11).					
<virtAntIdx 6>	<virtAntIdx 7>	<virtAntIdx 8>	<virtAntIdx 9>	<virtAntIdx 10>	<virtAntIdx 11>
int8	int8	int8	int8	int8	int8
ISK : -4 ODS : -3 AOP : -2	ISK : -5 ODS : -3 AOP : -2	ISK : -4 ODS : -2 AOP : -1	ISK : -5 ODS : -2 AOP : -1	ISK : -6 ODS : -3 AOP : 0	ISK : -7 ODS : -3 AOP : 0
Location index of each virtual antenna in the azimuth domain (see Figure 11).					

antGeometry1					
<virtAntIdx 0>	<virtAntIdx 1>	<virtAntIdx 2>	<virtAntIdx 3>	<virtAntIdx 4>	<virtAntIdx 5>
int8	int8	int8	int8	int8	int8
ISK : -1 ODS : 0 AOP : -1	ISK : -1 ODS : -1 AOP : 0	ISK : -1 ODS : -1 AOP : -1	ISK : -1 ODS : 0 AOP : 0	ISK : 0 ODS : 0 AOP : -3	ISK : 0 ODS : -1 AOP : -2
Location index of each virtual antenna in the elevation domain (see Figure 11).					
<virtAntIdx 6>	<virtAntIdx 7>	<virtAntIdx 8>	<virtAntIdx 9>	<virtAntIdx 10>	<virtAntIdx 11>
int8	int8	int8	int8	int8	int8
ISK : 0 ODS : -1 AOP : -3	ISK : 0 ODS : 0 AOP : -2	ISK : -1 ODS : -2 AOP : -3	ISK : -1 ODS : -3 AOP : -2	ISK : -1 ODS : -3 AOP : -3	ISK : -1 ODS : -2 AOP : -2
Location index of each virtual antenna in the elevation domain (see Figure 11).					

In the IWR6843 ISK antenna pattern, antGeometry1 indicates two elevation rows (specified by 0 and -1), and antGeometry0 indicates the virtual antenna's physical locations along the azimuth axis (specified by 0, -1, -2, -3, …, -7) on each elevation row, as shown in Figure 10a and Figure 11a.

In the IWR6843 ODS/AOP antenna patterns, antGeometry1 indicates four elevation rows (specified by 0, -1, -2, -3), and antGeometry0 indicates the physical location of the virtual antenna along the azimuth axis (specified by 0, -1, -2, -3) on each elevation row. Figure 10b and Figure 11b present the index positions of the ODS pattern. Similarly, the index positions of the AOP pattern can be found in Figure 10c and Figure 11c.

3.3.2. Antenna Phase Rotation Configuration

antPhaseRot					
<virtAntIdx 0>	<virtAntIdx 1>	<virtAntIdx 2>	<virtAntIdx 3>	<virtAntIdx 4>	<virtAntIdx 5>
int8	int8	int8	int8	int8	int8
ISK : 1 ODS : 1 AOP : 1	ISK : 1 ODS : -1 AOP : -1	ISK : 1 ODS : -1 AOP : 1	ISK : 1 ODS : 1 AOP : -1	ISK : 1 ODS : 1 AOP : 1	ISK : 1 ODS : -1 AOP : -1
Phase rotation of each virtual antenna.					
<virtAntIdx 6>	<virtAntIdx 7>	<virtAntIdx 8>	<virtAntIdx 9>	<virtAntIdx 10>	<virtAntIdx 11>
int8	int8	int8	int8	int8	int8
ISK : 1 ODS : -1 AOP : 1	ISK : 1 ODS : 1 AOP : -1	ISK : 1 ODS : 1 AOP : 1	ISK : 1 ODS : -1 AOP : -1	ISK : 1 ODS : -1 AOP : 1	ISK : 1 ODS : 1 AOP : -1
Phase rotation of each virtual antenna.					

This parameter defines the phase rotation introduced in the board design. Each field should be set to 1 if no rotation exists and -1 if there is a phase rotation in the corresponding antenna element. As depicted in Figure 10a, all the antenna elements of the ISK pattern are fed from the same direction. Hence, the phase rotation for each virtual element of the ISK pattern is configured to 1. On the other hand, as shown in Figure 10b and Figure 10c, some virtual elements of the ODS and AOP patterns are fed from the opposite side as compared to others. Therefore, the phase rotation of 180 degrees (i.e., -1) is applied to these elements, as summarized in the table above.

3.3.3. Range Bias and Phase Compensation Configuration

compRangeBiasAndRxChanPhase					
<rangeBias>	<virtAntIdx 0>	<virtAntIdx 1>	<virtAntIdx 2>	<virtAntIdx 3>	<virtAntIdx 4>
N/A	float, float	float, float	float, float	float, float	float, float
0	1, 0	1, 0	1, 0	1, 0	1, 0
Range bias for the array. Always set to 0. This field is not used in the chain.	Phase compensation factor (real, imaginary) of each virtual antenna.				
<virtAntIdx 5>	<virtAntIdx 6>	<virtAntIdx 7>	<virtAntIdx 8>	<virtAntIdx 9>	<virtAntIdx 10>
float, float	float, float	float, float	float, float	float, float	float, float
1, 0	1, 0	1, 0	1, 0	1, 0	1, 0
Phase compensation factor (real, imaginary) of each virtual antenna.					
<virtAntIdx 11>	-	-	-	-	-
float, float	-	-	-	-	-
1, 0	-	-	-	-	-
Phase compensation factor (real, imaginary) of each virtual antenna.					

This set of range and phase compensation parameters can be derived from TI's mmWave-SDK out-of-box (OOB) demo [5] (the “Range Bias and Rx Channel Gain/Phase Measurement and Compensation” section of the xwr68xx OOB demo). Please refer to the procedure defined in mmWave-SDK for how to obtain them. If the user decides not to compensate for the bias (which is not suggested), the default value in the above table can be set in the configuration file. The results from the OOB demo is always in the order of:

- [TX0-RX0, TX0-RX1, TX0-RX2, TX0-RX3, TX1-RX0, TX1-RX1, TX1-RX2, TX1-RX3, TX2-RX0, TX2-RX1, TX2-RX2, TX2-RX3],

which indicates that the TDM-MIMO needs to follow the default order of TX0, TX1, and TX2 as discussed in Section 3.1.3:

- chirpCfg 0 0 0 0 0 0 0 1
- chirpCfg 1 1 0 0 0 0 0 2
- chirpCfg 2 2 0 0 0 0 0 4

If the user configures the chirp with a different TX order, then it is the user's responsibility to shuffle the OOB calibration results as well as antGeometry0, antGeometry1, and antPhaseRot to match with the new visual antenna order.

3.3.4. Field-of-View (FOV) Configuration

fovCfg			
<subFrameIdx>	<azimuthFoV>	<elevationFoV>	-
int8	float	float	-
-1	3DWM : 70 3DCM : 60	3DWM : 20 3DCM : 60	-
Subframe index. Always set to -1.	Azimuth FOV. Assuming that the ISK is used in 3DWM and ODS/AOP is used in 3DCM.	Elevation FOV. Assuming that the ISK is used in 3DWM and ODS/AOP is used in 3DCM.	-

This command defines the sensor azimuth and elevation FOVs. The FOV values in this command define the total angular extent observed at one side of the sensor. In other words, a value of θ in this command configures the corresponding FOV to lie in the closed interval $[-\theta, \theta]$. Targets outside of this angular FOV interval are not detected in the people tracking demo.

Please note that these parameters also have a big impact on the memory and processor usage depending on the angular search granularities configured in dynamicRangeAngleCfg and dynamic2DAngleCfg (see Section 3.2). The current build of the implementation only supports the maximum FOV defined in the table above for the suggested angular search grids. The user needs to adjust the local heap memory to be able to support a wider FOV. If the user configures a wider FOV without changing the mentioned angular search granularities, then it is the user's responsibility to optimize the memory and processing time accordingly.

4. Parameter Tuning and Performance

This section guides the user to tune the configuration parameters discussed in the previous sections and presents how the detection layer performance is affected when these parameters are changed.

4.1. Effect of the Radar Parameters on the Physical Requirements

First, we detail the physical parameters in real-world scenarios and their relations to the radar system parameters discussed in this document. In a real-world scenario, one of the most critical limitations that the user should consider in their applications is the sensor's ability to distinguish between targets that are very close in either range, velocity, or angle domains, which is typically known as resolution:

- **Range resolution:** The range resolution defines the minimum separation in the range domain at which the radar can distinguish two targets and detect them as separate objects.
- **Velocity resolution:** The velocity resolution is the minimum radial velocity difference between two objects for the radar to detect them as separate objects in the Doppler domain.
- **Angle resolution:** The angle (i.e., azimuth and elevation) resolution defines the minimum separation in the angular domain at which the radar can distinguish two targets and detect them as separate objects.

Besides, the following parameters are also essential to define the physical extent of the target scene:

- **Maximum unambiguous range:** This parameter specifies the maximum target range that the radar can unambiguously resolve. Targets at ranges beyond the maximum unambiguous range r_{max} can not be detected because only the range interval $[0, r_{max}]$ is available at the range-FFT output.
- **Maximum detection range based on SNR:** The maximum range at which a human can be detected. This parameter is computed by the link budget formula discussed below, which is a function of the detection SNR, radar cross-section of the object, RF performance of the radar device, antenna gains, and the chirp parameters.
- **Maximum unambiguous velocity:** This parameter specifies the maximum magnitude value of the target's radial velocity that the radar can unambiguously resolve. Targets detected at velocities whose magnitude is greater than the maximum unambiguous velocity v_{max} are wrapped into the interval $[-v_{max}\ v_{max}]$. In other words, objects moving faster than this value may have incorrect velocity measurements.

Table 3 summarizes the radar system parameters that govern the physical bounds of the processing chain presented above. Besides, it provides the corresponding values of each parameter for both 3D wall-mount and ceil-mount scenarios. Figure 12 illustrates the chirp parameters that govern the physical performance limits discussed in this section. To get a detailed understanding of the radar system parameters, please refer to the mmWave training series in [4] and the application report in [8].

It is important to emphasize that the bandwidth (B) and chirp time (T_c) refer to the valid sweep bandwidth and ADC sampling time, respectively, discussed in Section 3.1.2 (see Figure 6). The chirp repetition period (T_r) is $N_{\text{TX}} \times (T_{\text{idle}} + T_{\text{ramp}})$, where N_{TX} is the number of TX antennas, T_{idle} is the idle time and T_{ramp} is the ramp end time in Figure 6 configured by profileCfg->idleTime and profileCfg->rampEndTime, respectively, in Section 3.1.2.

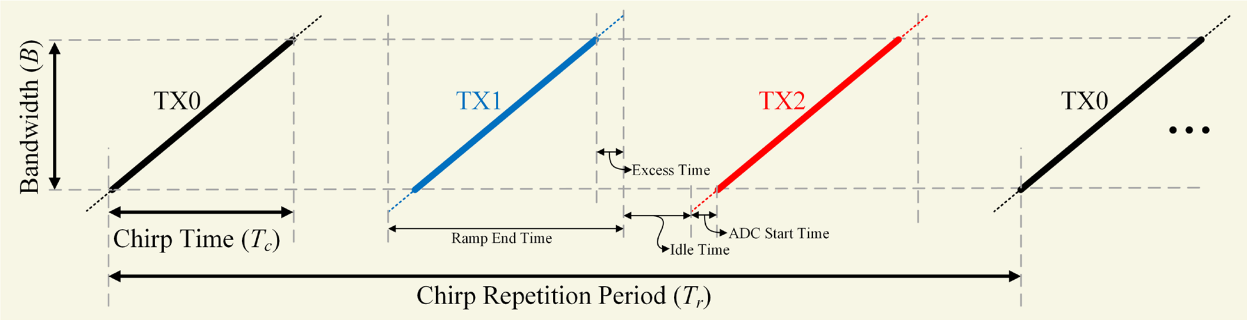


Figure 12. The chirp parameters that govern the physical performance limits.

The center frequency f_c in Table 3 is computed as

$$f_c = \underbrace{f_0 + T_{\text{ADC}}K}_{\hat{f}_0} + B/2, \qquad \hat{f}_0\text{: effective start frequency} \tag{2}$$

where f_0 is the start frequency configured by profileCfg->startFreq, T_{ADC} is the ADC start time configured by profileCfg->adcStartTime, and K is the frequency slope configured by profileCfg->freqSlope in Section 3.1.2.

The maximum beat frequency f_b in Table 3 depends on the ADC sampling frequency f_s configured by profileCfg->digOutSampleRate in Section 3.1.2. In a complex 1x sampling mode supported by the people tracking demo (see adcCfg->adcOutputFmt in Section 3.1.1), the IF bandwidth is limited to is $0.9 \times f_s$ [8].

Finally, the following are the other physical parameters used in the given analysis:

Speed of light:	$c = 3e8\ m/s$
Wavelength:	$\lambda = c/f_c$
Boltzmann's constant:	$k = 1.38e - 23$
Ambient temperature:	$T_e = 290°K$
Radar cross-section:	$\sigma = 1\text{m}^2$
Transmit power:	$P_{TX} = -10\text{dBm}$
Combined TX/RX antenna gain:	$G_{TX}G_{RX} = 16\text{dB}$
Total system loss:	$L = 5\text{dB}$
Noise figure of the receiver:	$\eta = 16\text{dB}$
Required detection SNR:	$\text{SNR} = 12\text{dB}$
Total virtual antennas:	$N_{TX}N_{RX} = 12$

Table 3. The sensor and physical parameters of the 3D people tracking demo that governs the system performance.

Parameter	Unit	3D Wall-Mount	3D Ceil-Mount
Bandwidth, B	MHz	1780.41	1768.66
Chirp time, T_c	µs	32.5	32.0
Inter-chirp time, T_r	µs	267.30	330.00
Number of chirps per frame (per TX antenna), N_c	-	96	224
Maximum beat frequency, f_b	MHz	2.66	1.8
Center frequency, f_c	GHz	63.01	63.02
Maximum unambiguous range, $r_{max,u} = (cf_b)/(2K)$	m	7.28	4.89
Maximum detection range based on SNR, $r_{max,d} = \sqrt[4]{\frac{\sigma\, P_{TX}\, G_{TX} G_{RX}\, \lambda^2\, T_c N_c N_{TX} N_{RX}}{(4\pi)^3 k\, T_e\, \eta\, L\, \mathrm{SNR}}}$	m	18.27	22.48
Range resolution, $\delta_r = c/2B$	m	0.0842	0.0848
Maximum unambiguous velocity, $v_{max} = c/(4f_c T_r)$	m/s	4.45	3.61
Velocity resolution, $\delta_v = c/(2N_c f_c T_r)$	m/s	0.0928	0.0322

Please note that changing any of these detection layer parameters may result in the system running out of memory and MIPS. The provided values have been carefully tuned for the best performance with the current device memory and MIPS limitations. For more details, please refer to the implementation guide in [1].

4.2. How to Configure the Range

This section lists all the parameters that affect the range measurements and presents the relation of each parameter with the range boundary and resolution in real-world scenarios.

4.2.1. FMCW Chirp Parameters

In the following table, we first provide the relationships among the system's range requirement and the FMCW chirp waveform parameters discussed in Section 3.1.2.

Commands	Parameters	Effect on the Range Extension
profileCfg	<freqSlope>	This parameter affects the bandwidth. Therefore, it manages both the range resolution and maximum unambiguous range. Increasing the frequency slope improves the range resolution but decreases the maximum unambiguous range. The user must ensure that the occupied bandwidth is within the sensor (i.e., 60 to 64 GHz) and the region-based regulation (FCC, etc.) limits.
	<numAdcSamples>	This parameter affects the chirp time and bandwidth. When the frequency slope and the sampling rate are constant, it manages the range resolution. Increasing the number of ADC samples results in a wider bandwidth that improves the range resolution. This parameter also affects the valid chirping time and maximum detectable range based on SNR. On the other hand, this parameter affects the radar cube size and required processing power. Changing this parameter may result in the system running out of memory and MIPS [1].
	<digOutSampleRate>	This parameter affects the chirp time and bandwidth. When the frequency slope and the number of ADC samples are constant, it manages both the range resolution and maximum unambiguous range. In this case, increasing the sampling rate results in a narrower bandwidth that degrades the range resolution but increases the maximum unambiguous range. This parameter also affects the valid chirping time and maximum detectable range based on SNR.

4.2.2. Transmit Power

As discussed in Section 3.1.2, TI's IWR6843 mmWave sensors have a backoff capability to reduce the transmit power from the maximum that can be tuned according to the detection range requirements of different use-cases. If the transmit backoff is reduced to 0dB (corresponds to the maximum 12dBm transmit power), the user can improve the detection range, as discussed in Section 4.1. The suggested values in this document are empirically adjusted to achieve a better detection performance both for wall-mount and ceil-mount scenarios. Configuring the power as too high (i.e., using a backoff value close to 0dB) may cause stronger multipath reflections and higher false alarms. On the other hand, too low TX power will decrease the detection range performance, which may result in more missed detections.

Commands	Parameters	Effect on the Range Extension
profileCfg	<txOutPowerBackoff>	This parameter affects the maximum detection range and should be configured based on the use-case and regulation requirements.

4.2.3. CFAR Parameters

As discussed in the previous sections, the 1D CFAR-CASO algorithm is applied across the range and angle (if the second-pass is enabled, as discussed in Section 3.2.2) domains for target detection. As shown in the CFAR-CASO mechanism for a 1D data set in Figure 13, the noise power is estimated as the average power of the reference cells within a configured window around each cell under test. The CFAR-CASO detector assumes that the reference cells do not contain any signals from targets. Therefore, guard cells prevent noise estimation from signals leaking into the reference cells from the target peaks.

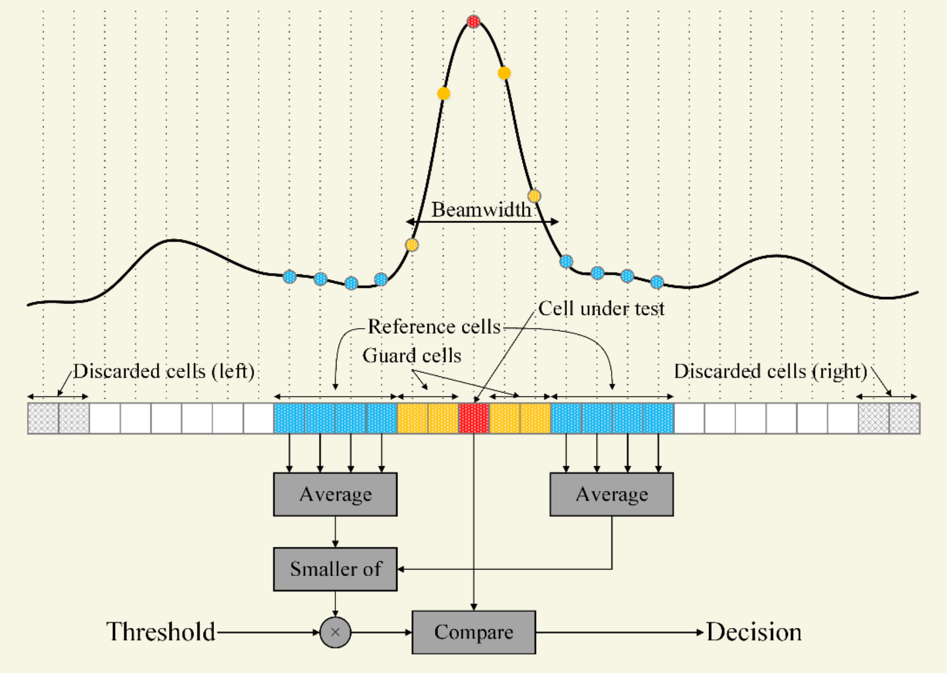


Figure 13. The 1D CFAR-CASO detector estimates the noise variance for the cell under test from the neighboring reference cells.

The reference and guard window sizes in Figure 13 significantly affect detection performance and should be carefully tuned according to the range parameters configured above. As discussed in Section 3.2.2 and illustrated in Figure 13, both window sizes and the discarded cells on both left and right sides are configured in terms of the number of samples. Hence, depending on the chirp configuration and derived inter-bin range resolution, the user must accordingly tune the following parameters. For example, a higher chirp bandwidth results in a sharper peak in the range FFT output, and therefore, the guard window size can be reduced to improve the noise estimation accuracy. Similarly, when the number of ADC samples are or the sampling rate change, the user should adjust the reference and guard cell sizes accordingly.

Commands	Parameters	Effect on the Range
dynamicRACfarCfg	<cfarDiscardLeftRange>	These parameters are defined in the number of samples. Therefore, when the range resolution and the number of ADC samples are changed, these parameters should be adjusted accordingly.
	<cfarDiscardRightRange>	
	<refWinSizeRange>	
	<guardWinSizeRange>	
staticRACfarCfg	<cfarDiscardLeftRange>	Similar to the dynamic counterparts, these parameters should be adjusted accordingly when the range resolution and the number of ADC samples are changed.
	<cfarDiscardRightRange>	
	<refWinSizeRange>	
	<guardWinSizeRange>	

4.3. How to Configure the Motion Sensitivity

In the 3D people tracking demo, accurate knowledge of radial velocity is essential to improve tracking performance. More importantly, optimizing the motion sensitivity (i.e., the velocity resolution) is a crucial task in the 3D people tracking demo to be able to distinguish between the purely static objects (such as chairs, tables, etc.) and the people who remain stationary or with very slow motion in the scene. As discussed in Section 2, to successfully leave the signals scattered from the moving objects after the static clutter removal step, a high-velocity resolution is needed to improve the motion sensitivity. On the other hand, when the velocity resolution is increased, the processing chain becomes more sensitive to undesirable motion artifacts such as sensor vibration. The user should consider a proper tradeoff between the detection and false alarm performance according to the target environment.

As presented in Section 4.1, the increased total chirp observation window consists of multiple successive chirps, and the inter-chirp time affects the maximum unambiguous velocity and velocity resolution. This section provides a tuning approach for the system's velocity requirements configuring the related radar parameters.

Commands	Parameters	Effect on the Motion Sensitivity
profileCfg	<idleTime>	These parameters affect the inter-chirp time (i.e., T_r in Section 4.1), hence both the velocity resolution and maximum unambiguous velocity. When the number of chirps per frame is the same, increasing the inter-chirp time improves the velocity resolution but decreases the maximum unambiguous velocity. Therefore, the people tracking demo becomes more sensitive to the motion. It is important to note that increasing the inter-chirp time through these parameters increases the total chirping window and reduces the total available processing time budget within the frame period. Hence, the user must ensure that the total chirping time and the required processing time [1] can fit into a single frame period, which is configured by frameCfg-> framePeriodicity.
	<rampEndTime>	
frameCfg	<numLoops>	This parameter (i.e., N_c in Section 4.1) affects the total chirping window. Increasing the number of chirp loops improves the velocity resolution but also reduces the total available processing time budget within the frame period. Hence, the user must ensure that the total chirping time and the required processing time [1] can fit into a single frame period, configured by frameCfg-> framePeriodicity. On the other hand, this parameter affects the radar cube size and required processing power. Changing this parameter may result in the system running out of memory and MIPS [1].

4.4. How to Configure the Angular Accuracy

The FOVs of both ODS/AOP and ISK EVMs are discussed in Section 3.3.4. Based on the sensor mounting (i.e., wall or ceil) option and the required target scene boundary, the user can adjust these angular limits accordingly. However, it is important to emphasize that the user should consider the antenna radiation patterns' physical limitations when configuring these parameters.

The angle resolution described in Section 4.1 is imposed by the total number of antenna elements used in the sensor array. As discussed in Section 3.3.1, the ODS/AOP EVMs antenna pattern has four elements in both azimuth and elevation domains. The ISK EVM antenna pattern has eight elements in the azimuth domain and only two elements in the elevation domain. Despite the physical limitations in the angle resolution, the angle estimation accuracy can be improved by increasing the angular search grid in the Capon beamformer's steering matrix computations.

This section provides a tuning approach for the angular accuracy requirements by configuring the related radar parameters.

Commands	Parameters	Effect on the Range Extension
dynamicRangeAngleCfg	<angleSearchStep>	When the dynamicRangeAngleCfg->rangeAngleEstMethod is configured to 1, this parameter configures the search resolution in the azimuth domain. When the dynamicRangeAngleCfg->rangeAngleEstMethod is configured to 2, this parameter configures the coarse angle search resolution in both azimuth and elevation domains, which will be zoomed-in, as discussed in Section 3.2.3. A smaller value will improve the estimated azimuth/elevation accuracy but also increases the total number of search angles if the fovCfg->azimuthFoV or fovCfg->elevationFoV is not changed. The user should make a tradeoff between the search resolution and the FOV and adjust the angle search steps according to the memory and computation power limits. Changing this parameter may result in the system running out of memory and MIPS [1]. To fit the required processing time into a single frame period, the user must configure the frame period through frameCfg-> framePeriodicity.
dynamic2DAngleCfg	<elevSearchStep>	When the dynamicRangeAngleCfg->rangeAngleEstMethod is configured to 1, this parameter configures the search resolution in the elevation domain. A smaller value will improve the estimated elevation accuracy but also increases the

		total number of search angles if the fovCfg->elevationFoV is not changed. The user should make a tradeoff between the search resolution and the FOV to adjust the elevation search step based on the memory and computation power limits. Similar to the angleSearchStep parameter above, changing this parameter may result in the system running out of memory and MIPS [1].
	<zoominFactor>	When the dynamicRangeAngleCfg->rangeAngleEstMethod is configured to 2, this parameter configures the fine angle search resolution in both azimuth and elevation domains. A larger zoom-in factor will improve the estimated azimuth and elevation accuracy but also increases the required memory and MIPS. Similar to the angleSearchStep parameter above, changing this parameter may result in the system running out of memory and MIPS [1].
staticRangeAngleCfg	<staticAzimStepDeciFactor>	As discussed in Section 3.2.4, these parameters configure the angular search resolution in both azimuth and elevation domains when estimating the static scene. Based on the azimuth and elevation search granularities configured for the dynamic scene, A smaller decimation factor will improve the estimated azimuth and elevation accuracy of the static points but also increases the required memory and MIPS. Changing this parameter may result in the system running out of memory and MIPS [1].
	<staticElevStepDeciFactor>	

Similar to the discussion in Section 4.2.3, the reference and guard window sizes in the 1D angle-CFAR algorithm (if the second-pass is enabled) significantly affect detection performance and should be carefully tuned according to the angular grid parameters configured above. As discussed in Section 3.2.2 and illustrated in Figure 13, both window sizes and the discarded cells on both left and right sides are configured in terms of the number of samples. Hence, depending on the derived inter-bin angle resolution, the user must accordingly tune the following parameters.

Commands	Parameters	Effect on the Range
dynamicRACfarCfg **staticRACfarCfg**	<cfarDiscardLeftAngle>	These parameters are defined in the number of samples. Therefore, when the inter-bin angle resolution and the FOV are changed, these parameters should be adjusted accordingly.
	<cfarDiscardRightAngle>	
	<refWinSizeAngle>	
	<guardWinSizeAngle>	

5. References

[1] 3D people tracking demo implementation details', Texas Instruments.

[2] Group tracker implementation guide, Texas Instruments.

[3] Group tracker tuning guide, Texas Instruments.

[4] MmWave training series, Texas Instruments. [Online]. Available: https://training.ti.com/mmwave-training-series

[5] MmWave software development kit, Texas Instruments. [Online]. Available: https://www.ti.com/tool/MMWAVE-SDK

[6] MmWave sensors industrial toolbox, Texas Instruments. [Online]. Available: https://dev.ti.com/tirex/explore/node?node=AJoMGA2ID9pCPWEKPi16wg__VLyFKFf__LATEST

[7] Sandeep Rao, "MIMO radar," Texas Instruments, Application Report, SWRA554A, July 2018.

[8] Vivek Dham, "Programming chirp parameters in TI devices," Texas Instruments, Application Report, SWRA553A, February 2020.

6. Appendix

6.1. Template of Parameter Definition Tables

In Section 3, the configuration parameters are explained in detail for each parameter set. All the parameters belong to the corresponding parameter set are presented in separate tables according to the template in Table 4.

Table 4. Template of parameter definitions of an example command.

configurationParameterSet (an example command)			
<parameter1Label>	< parameter2Label >	< parameter3Label >	< parameter4Label >
Data type			
Suggested values for: • 3D Wall-Mount (3DWM) • 3D Ceil-Mount (3DCM)			
Parameter descriptions and important notes.			
< parameter5Label >	< parameter6Label >	< parameter7Label >	< parameter8Label >
Data type			
Suggested values for: • 3D Wall-Mount (3DWM) • 3D Ceil-Mount (3DCM)			
Parameter descriptions and important notes.			