Halmstad University

FMCW radars

JACCOD Emma - SOW Aicha N.

June 2021

Internship supervisors : ANDREASSON Pererik & FRIEL Ross

Table des matières

Table des figures

1 Glossary

ADC : Analog to Digital Converter.

API : Application programming interface.

Beat Frequency Beat frequency is the difference between the original ramp signal and the echo reflection from the detected object.

CFAR : To reliably detect objects, constant detection thresholds are to be avoided.
Instead, Constant False Alarm Rate (CFAR) algorithms are used to compute adaptive threshold depending on the estimated noise floor, clutter and interference

Chirp : Signal emitted by radar module.

Frame : A sequence of chirps.

CW : Continuous Wave.

FMCW : Frequency Modulated Continuous Wave.

DOA : Direction Of Arrival.

DSP : Digital Signal Processor.

EM : Electromagnetic.

FFT : Fast Fourier Transform.

FoV : Field of View.

Frequency Slope : Slope of the chirp as it ramps frequency. Frequency Slope must be equal to or less than 100 MHz/us.

Frequency Start : Starting frequency of the chirp ramp, which determines the carrier wavelength λ.

SNR : Signal to noise ratio.

SDK : Software development kit.

TLVs : data structured with the TLV (Type-Length-Value) format.

Gtracker : Group tracker algorithm.

TI : Texas Instruments.

GUI : Graphical User Interface.

MIMO radar : Multiple-input and multiple-output radar.

2 Radar Theory

RAdio Detection And Ranging or radar is an electromagnetic [EM] sensor, used to notice, track, locate, and identify different objects or targets.

Radars produce EM signals that are radiated into space thanks to the Tx Antennas.
When a transmitted signal reaches an object, it gets reflected or re-radiated (refraction) in one or many directions. The reflected signal (or echo signal), once received by the radar's Rx antenna is amplified, down-converted to an intermediate frequency [IF] and processed to determine the geographical statistics of detected objects.

The raw radar data can be composed of a cosine wave-form (I signal) **or** a sine wave-form (Q signal) **or** both.
These signals have the same amplitude and frequency but their phases are shifted 90° with respect to each other.

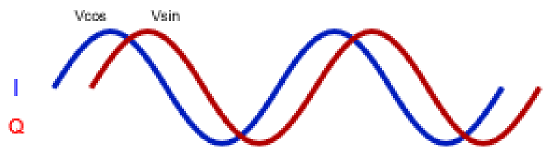


FIGURE 1 – I and Q signals

I (or In-phase) and Q (or Quadrature) demodulators are often used in radars to dechirp the Radar's received signal.
IF the mixer outputs only real samples (I -only or Q-only), only the absolute frequency shift can be obtained making it impossible to know the direction of travel of the target. While if both **I Q samples** (complex mixer) are outputted, the direction of travel can be known by checking which of the signals is leading in phase.

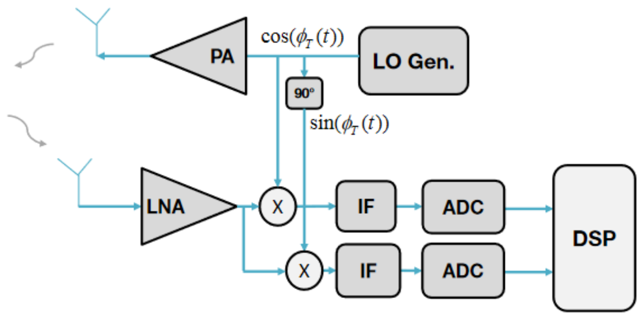


FIGURE 2 – The complex complex-baseband architecture. Image source : [8]

There are different types of radar systems that are classified according to their functions,their transmission rate, the geometry of their antennas and the illuminator they use : unmodulated continuous-wave radars, modulated continuous-wave radars, Doppler radars, etc.

In this study, we will focus on FMCW radars as the radar board used in this project is one of them.

2.1 Principle of Frequency Modulated Continuous Waves

FMCW radars, just like other CW radars continuously transmit and receive (simultaneously) wave signals. But in contrast to non-modulated CW radars, FMCW radars can change their operating frequency during measurements as the transmission signal is modulated in frequency. The permanent variation of the operating frequency allows the detection of fixed targets and moreover, certain measurements (i.e range) are only technically possible with these changes in the frequency (or phase).

The echo signal received by the radar is a time-delayed copy of the transmitted signal where the delay, τ, is related to the range of the target refracting the signal.

As the signal is always changing in frequency, at any given moment during the sweep through the frequency band, the frequency difference or beat frequency, f_b, is a constant between the transmitted signal and the received signal.

There are different forms of frequency modulation but the most famous one is linear frequency modulation. In this case, because the sweep is linear, we can derive the time delay (τ) from the beat frequency f_b and from there determine the range r of the reflecting object.

$$range\ r = \frac{C * \tau}{2} = \frac{C * f_b}{2 * (\frac{df}{dt})}$$

df/dt : frequency shift per unit of time

c : speed of light

A variety of modulation patterns can be used for different measurement purposes : sawtooth modulation, triangular modulation, square-wave modulation, etc. see [5] For instance, the sawtooth modulation is presented in the figure below :

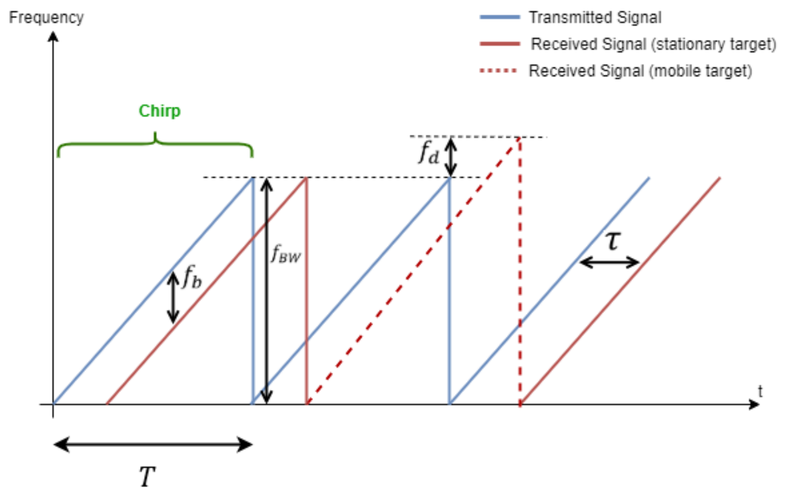

Frequency
Transmitted Signal
Received Signal (stationary target)
Received Signal (mobile target)
Chirp
f_d
f_b
f_{BW}
τ
t
T

FIGURE 3 – FMCW with sawtooth modulation .

f_b : beat frequency : the frequency difference between the transmitted and received signal. It is also referred to as the intermediate frequency (IF)

f_D : Doppler shift

f_{BW} : modulation bandwidth.

T : modulation period or sweep time.

Note : the full line represents the signal reflected by a stationary target and the dotted line, the one reflected by a mobile target.

The signal emitted by the radar module during its modulation period is called a Chirp. With a set of chirps composing a Frame. See figure 2 below.

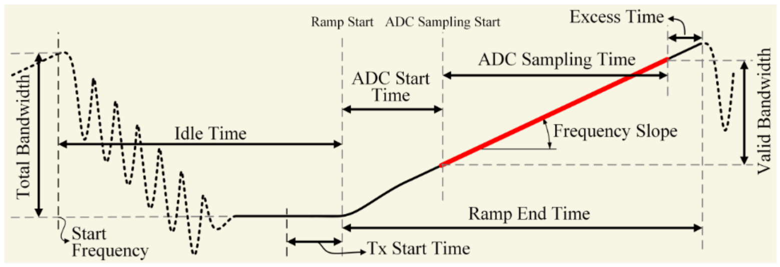


FIGURE 4 – Chirp signal parameters. Figure by TI

For every chirp, multiple ADC samples can be made.

2.1.1 Stationary target

For a stationary target supposed at a distance d, delay between received signal and transmitted signal is $\tau = \frac{2*d}{c}$, with c being the speed of light. This delay is due to the signal traveling between the target and the radar.

In the case of a stationary target, the beat frequency is mainly due to the target's distance from the radar.
Indeed, there is no Doppler effect as the target is not moving.

Thus, the following results are obtained :

$$\frac{f_{BW}}{T} = \frac{f_b}{\tau} \text{ with } \tau = \frac{2*d}{c} \rightarrow f_b = f_{BW}\frac{2*d}{c*T}$$

2.1.2 Mobile target

For mobile targets, the Doppler effect needs to be kept in mind because of the relative motion between the target and the radar.

Indeed, the distance will be affected by signal compression or elongation depending on the object movement. The beat frequency now depends on the time delay with the addition of Doppler shift (f_D).

Thus,

$$f_b = f_{BW} + f_D \text{ with } f_D = 2 * f_0 * \frac{v}{c}$$

To obtain the velocity and angle information of different objects, it is necessary to do three FFT (Fast Fourier Transform)

For that it is necessary to start from a matrix containing all the chirps (each column for the beat signal of a single chirp).

The first FFT is useful to obtain the range. The second one, the velocity. The last one, the angle of objects.

Indeed, the Fourier transform provides a frequency spectrum. On this spectrum there are some peaks and for each peak there is one object at a specific distance.

Thus, through FFT, it is possible to obtain the range of different objects. This first FFT is the "*range-FFT*".

However, there can be different peaks at same location but with different phases. The difference in phase between peaks is due to the spatial motion of the objects. If there are different objects at the **same distance** from the radar but with **different velocities**, Range FFT will not be enough to tell them apart.

To solve this issue, a *Doppler-FFT* is needed. This one is an FFT on the phase sequence.

Now, if there are different objects at the **same distance** from the radar and with the **same velocity**, we need to use the angle of arrival for differentiation.
Angle estimation is based on the phase change in the peak of the range-FFT or Doppler-FFT because of differential distance from the object to each of the antennas.

Hence, at least two Rx antennas (for two objects) are useful for accurate detection in a "same-range, close velocity" case . This FFT is called "*angle-FFT*".

Further explanation for the FFT process can be found in this thesis : [6].

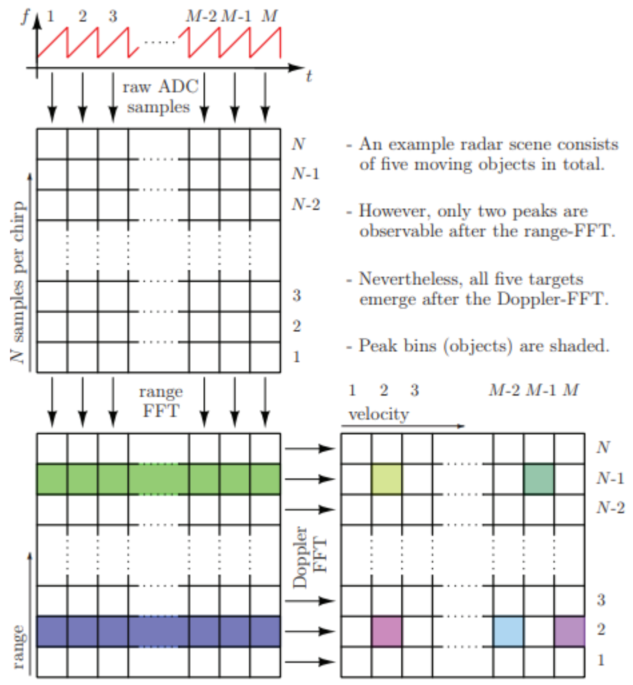

f
1 2 3 M-2 M-1 M
t
raw ADC samples
N samples per chirp
N
N-1
N-2
3
2
1
- An example radar scene consists of five moving objects in total.
- However, only two peaks are observable after the range-FFT.
- Nevertheless, all five targets emerge after the Doppler-FFT.
- Peak bins (objects) are shaded.
range FFT
range
Doppler FFT
1 2 3 M-2 M-1 M
velocity

FIGURE 5 – Process of the different FFTs. Image source [6]

In this example, there are M chirps and N samples per chirp. Thus, the first matrix contains $M \times N$ data. After the range-FFT, we observe two colors which correspond to two peaks for two different frequencies. Hence, all the objects are concentrated in two different locations. However, this doesn't mean there are only two objects. Indeed as shown below, objects can be located at the same distance but not the exactly same place :

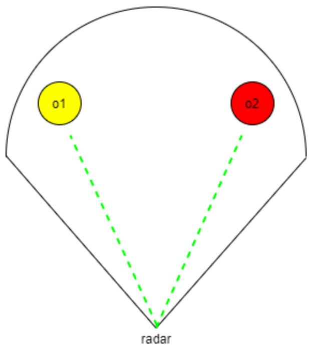


FIGURE 6 – Equidistant objects.

Thus, Doppler FFT is essential to distinguish equidistant objects. In the example above from the thesis [6] we can see that three objects were hidden under the first frequency and two under the second one, thanks to Doppler FFT,

2.2 Principle of measurements : basic radar equations

Given applications can have different requirements when it comes to range, range resolution, velocity, the processor MIPs or the amount of usable data memory.

We will see in the following subsections that a change in the chirp configuration (frequency slope,sweep bandwidth,etc) can influence the radar's performances.

Therefore, understanding the relationships between the chirp configuration and the system's performance metrics is crucial in order to suitably configure the chirp parameters to meet the requirements.

2.2.1 Signal to noise ratio

The signal to noise ratio, SNR compares the level of a desired signal to the level of background noise. The higher the SNR is , the less noise there is in the signal, as the formula below shows :

$$\text{SNR} = 10 * \log \frac{s^2(t)}{n^2(t)}$$

$s(t)$: desired signal

$n(t)$: background noise

2.2.2 Maximum radar range

The maximum radar range is the maximum distance at which a radar can still detect an object. To determine maximum radar range, the IF bandwidth (also referred to as beat frequency) and the slope of the transmitted chirp (S) can be used.

For Texas Instrument's AWR6843 radar, the IF bandwidth depends on the sampling mode of the ADC.

Thus, in TI's complex 1x sampling mode,

$$\text{max IF bandwidth} = 0.9 * (ADCsamplingRate)$$

and in a complex 2x and real sampling mode,

$$\text{max IF bandwidth} = \frac{0.9 * (ADCsamplingRate)}{2}$$

As the EM signal sent by the radar travels at the speed of light (C), the maximum distance is as follows :

$$d_{max} = \frac{IF\,bandwidth * C}{2 * S}$$

The measurable distance at different angles would also depend on the antenna's gain as it differs relatively to the angle of reception. The detection of a distant object can also be limited by the SNR of the received signal.[1]

2.2.3 Range resolution

A radar's range resolution is the minimum distance required to resolve two objects as separate entities rather than confuse them as one. This primarily depends on the chirp's sweep bandwidth. The larger the sweep bandwidth, the better (lower) the range resolution.

$$\text{Range resolution } \Delta\text{r} = \frac{C}{2 * B}$$

C : speed of light

B : Sweep bandwidth of the chirp

Note : The AWR6843 radar support a 4 GHz sweep bandwidth that allows a range resolution of approximately 4cm.

2.2.4 Angular Range and Resolution

Both the angle of the object and it's distance from the radar are needed in order to locate the object. In a radar system, the angle is estimated by receiving the reflected signal from the object using multiple receiver antennas that are spaced apart with a distance 'd'. The signal arriving at each of the successive receivers is delayed by $d * sin(\theta)$ causing a phase shift of :

$$\frac{2 * \pi * d * sin(\theta)}{\lambda}$$

This phase shift between each of the receivers is used to estimate the angle (θ) of the object.

The maximum unambiguous angular range is given by the following equation :

$$\arcsin\left(\frac{\lambda}{2 * d}\right)$$

Therefore, for the theoretically widest angular viewing range (±90°), the distance between receiver antennas should be $d = \frac{\lambda}{2}$. [1]

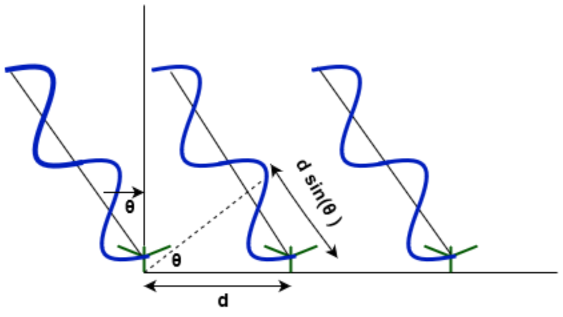


FIGURE 7 – Angle estimation.

Moreover, the angular resolution measurement depends on the number of receiver antennas available. The bigger the number of Rx antennas, the better the resolution.

$$\text{angular resolution in degrees} = \frac{\lambda}{d * Nrx * cos(\theta)} * \frac{180}{\pi}$$

θ : Angle of interest, angle at which the objects are present

Nrx : Number of receiver antennas

The use of multiple transmitters can also enhance (i.e lower) the angular resolution. In which case, the angular resolution takes into account the number of Tx antennas.

$$\text{angular resolution in degrees} = \frac{\lambda}{d * Nrx * Ntx * cos(\theta)} * \frac{180}{\pi}$$

Ntx : Number of transmit antennas

2.2.5 Maximum velocity and velocity resolution

The maximum measurable velocity in Fast FMCW radars depends on the chirp cycle time which in turn depends on how fast the frequency sweep can be performed and the minimum inter-chirp time allowed.

An object's velocity can be directly estimated from the measured phase difference of two peaks of chirp signals which corresponds to the spatial motion of the object :

$$\text{velocity} = \frac{\lambda * \Delta\phi}{4 * \pi * Tc} \text{ \& unambiguous max velocity} = \frac{\lambda}{4 * Tc}$$

$\Delta\phi$: The phase difference of two peaks of chirp signals

Tc : Total Chirp Time (chirptime + idletime)

λ : Wavelength of the signal

The maximum unambiguous velocity is the maximum range of radial velocity that can be observed without ambiguity by the radar. With the radial velocity being the motion along the radial (i.e directly toward or away from the radar) which is different from motion perpendicular to that radial or tangential velocity.

The velocity resolution that estimates how well two objects with small velocity differences can be resolved is obtained with the following equation :

$$\text{velocity resolution } \Delta\text{v} = \frac{\lambda}{2 * Nc * Tc}$$

Nc : number of chirps in a frame

Frame duration= Nc * Tc

The velocity resolution depends highly on the number of chirps in a frame and thus, on the transmit frame duration. Having more chirps in a frame improves the velocity resolution.

2.2.6 Power of the reflected signal

As we stated before, a radar's basic operation is to transmit an EM signal and receive the echo signal reflected by the target. The power of the echo signal can computed thanks to the radar equation :

$$\text{Prx} = \frac{Ptx * Grx * Gtx * \lambda^2 * \sigma}{(4 * \pi)^3 * R^4 * L}$$

Pt : Transmit power

Grx : Reception Antenna's gain

Gtx : Transmission Antenna's gain

λ : wavelength

σ : Target's radar cross section (RCS)

R : Range from the radar to the target

L : Losses due to propagation or/and system

Note : A target's RCS indicates how easily an object can be detected. It depends, for example, on the material of which the target is made or it's size. High RCS targets are deemed easier to detect.

TI's intoduction course on FMCW radars can be found here : [7]

3 TI mm-wave radar sensors

The millimeter-wave (mm-wave) radio frequency is commonly preferred over laser, ultrasound and other radio wave frequencies in industrial and automotive applications due to their robustness against bad weather conditions and harsh environments. For the labs, we used TI's AWR6843 radar sensor which is a Frequency Modulated Continuous-Wave (FMCW) radar capable of operating in the 57-64 Ghz range.

For software development, we use TI's mmWave SDK which is a collection of software packages that allows low and high level tweaking.

For that matter, we send a configuration file defining the wanted values of perfomance-bound parameters such as the number of chirp/frame, the number of Rx and Tx antennas to use and the raw analog radar signal sample rate.

(End Chirp TX index - Start Chirp Tx index) number of chirps constitute a sequence that will be repeated a number of Chirp Loop times to make a frame. The number of chirps per frame is then equal to (End chirp index + 1 - start chirp index) * number of chirp loops

The parameters of the chirps in a frame can be controlled by defining chirp profiles.

Indeed, 512 unique chirps can be pre-programmed and stored in the chirp configuration RAM.

TI's demo applications are built around two major subsystems : the MSS Master Sub-system which is also interchangeably referred to as Cortex R4F and the DSS DSP Sub-system or C674x core. The figure below shows the implementation of high end applications using the TI's mmwave sdk components.

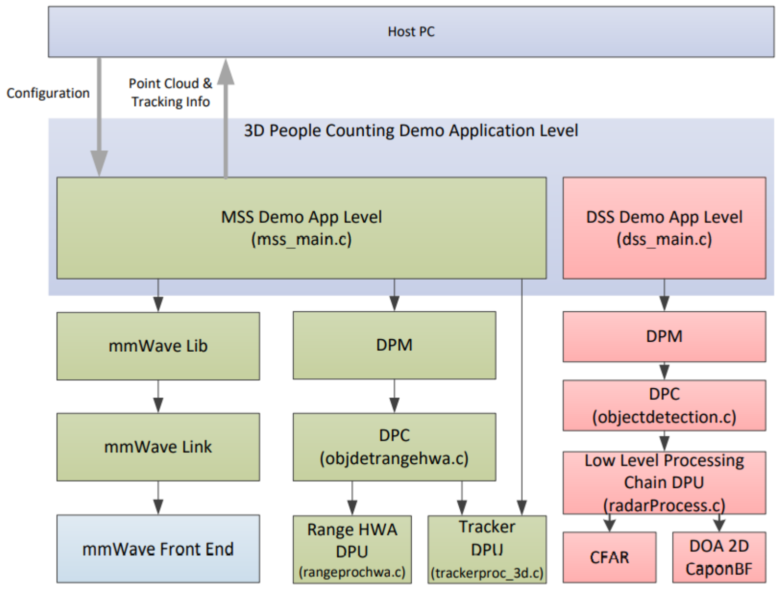

Host PC
Configuration
Point Cloud & Tracking Info
3D People Counting Demo Application Level
MSS Demo App Level (mss_main.c)
DSS Demo App Level (dss_main.c)
mmWave Lib
DPM
mmWave Link
DPC (objdetrangehwa.c)
mmWave Front End
Range HWA DPU (rangeprochwa.c)
Tracker DPU (trackerproc_3d.c)
DPM
DPC (objectdetection.c)
Low Level Processing Chain DPU (radarProcess.c)
CFAR
DOA 2D CaponBF

FIGURE 8 – implementation of TI's People counting demos. Image source : [3]

Please refer to [3] for further details regarding the mmwave sdk

3.1 People detection and tracking

The reflected radar signal or echo signal received by the Rx antennas are processed through multiple processing layers.

First we have the Front-End processing layer in which **ADC sampling** is made. Then, the ADC samples are processed in the **Detection layer** to produce a set of data points also referred to as Point Cloud.

Each point of the set represents a detected object with attributes such as its radial velocity, range, azimuth angle, SNR and spherical coordinates associated to it.
In fact, when an object is detected, it just means that the echo signal from that particular point had a high enough Signal to Noise Ratio to get through the radar's CFAR algorithm.
At this point, a point could be anything in the field of view of the radar. There is no concept of a Target or point of interest yet.

To be able to track people for instance, more post-processing needs to be done. To do so, the set of data points is fed to a Tracking layer which in our case is based on the GTracker algorithm.
The tracking layer localizes and tracks a clusters of points and outputs a list of targets. We will also use a classifier to tell apart people from still objects.

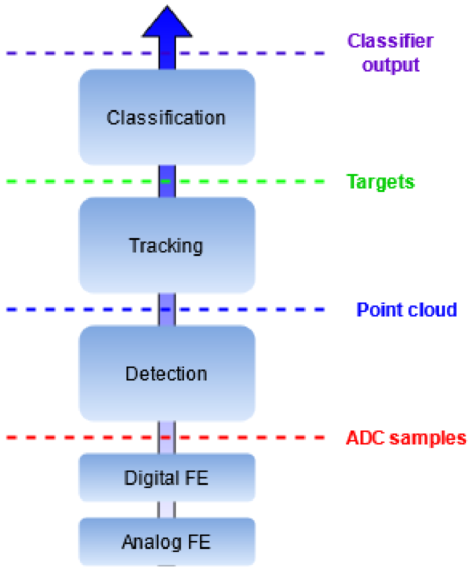


FIGURE 9 – Radar signal processing layers.

Note : The detection layer is implemented on a Hardware Accelerator (HWA) driven by the R4F cortex (ARM) processor , and the DSP (or C674x core). The Tracker Layer is implemented on the R4F.

3.2 Commonly used chirp configuration parameters

The FMCW chirp configuration is an important aspect to consider. That is why, in the chip configuration file, different parameters are given.

Each command which composes the chirp configuration is made by several numbers. Depending on the board and the demo, these numbers can change. For provided demos, the chirp configuration is supposedly made for the best performance. But improvement is possible.

In the following figures, the numbers shown serve only as examples. The default configuration file used for the 3D people detection application is available in subsection 3.3.1

3.2.1 defDataOutputMode command

dfeDataOutputMode command defines the ModeType :

dfeDataOutputMode 1
mode Type

Figure 10 – dfeDataOutputMode configuration.

In this example, the value given is 1 but it can also be 2 or 3 depends on the desired mode :

- **1 :** frame based chirps.
- **2 :** continuous chirping.
- **3 :** advanced frame config.

Noted that only 1 and 3 are supported for provided demo on this board.

3.2.2 channelCfg command

The second point is the channelCfg and as its name indicates, it corresponds to the channel configuration :

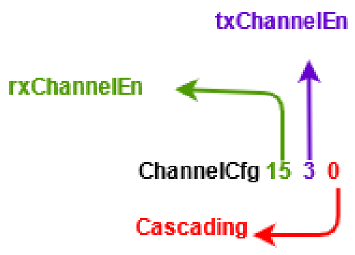


Figure 11 – Channel configuration.

- **rxChannelEn** corresponds to the receive antenna mask for the 4 antennas (for the AWR6843ISK) board). As all the receiving antennas should be switched-on, it is : 0x1111b = 15.
- **txChannelEn** corresponds to the transmit antenna mask for the 3 antennas (for the AWR6843ISK). Depends on the Tx antennas mask needed to enable the desired virtual antenna configuration, the value changes. Tx1 and Tx3 which are the 2 azimuth antennas can be enabled in the same time using bitmask 0x101b = 5. Also if all the 3 transmitting antennas are switched-on, the mask needs to be 0x111b= 7.

— **Cascading** corresponds to the SoC cascading mode. As it is not applicable here, set to 0.

3.2.3 adcCfg command

The adcCfg commande is for the ADC configuration :

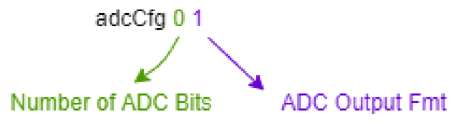


FIGURE 12 – adcCfg configuration.

— **Number of ADC Bits** is as its name indicates the Number of ADC bits :
  — 0 : for 12 bits
  — 1 : for 14 bits
  — 2 : for 16 bits
  For the AWR6843ISK, it has to be set to 2.
— **adcOutputFmt** corresponds to the Output Format :
  — 0 : real
  — 1 : complex 1x (image band filtered output)
  — 2 : complex 2x (image band visible)
  Noted that only complex modes are supported.

3.2.4 adcbufCfg command

The next point is the adcbufCfg which his the adcBuf hardware configuration described as below :

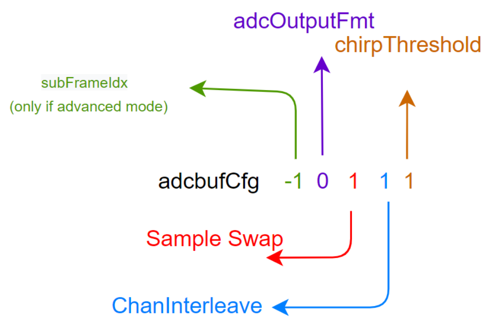


FIGURE 13 – adcbufCfg configuration.

— **subFrameIdx** is the subframe index. For advanced mode, it needs to be set to intended subframe number. As this last mode is not supported, this field will either be set to -1 or not filled at all depending on the demo.

— **adcOutputFmt** is the ADCBUF output format. The value taken is 0 for complex and 1 for real. For AWR6843, only complex modes are supported, set to to 0.

— **SampleSwap** is the ADCBUF IQ swap selection. Two values are possible :
  — 0 : I in LSB and Q in MSB

  — 1 : Q in LSB and I in MSB

  As only Q in LSB and I in MSB is supported, set ti to 1.

— **chanInterleave** is the ADCBUF channel interleave configuration. Two values are possible :
  — 0 : interleaved (only supported for xWR14xx).

  — 1 : non-interleaved.

  As only the non-interleaved option is supported, set it to 1.

— **ChirpThreshold** is the Chirp Threshold configuration used for ADCBUF buffer to trigger ping/pong buffer switch. Two cases are possible :
  — 0-8 : demos using DSP for FFT and LVDS streaming is disabled (xWR16xx demo).

  — 1 : demos using HWA for 1D FFT (xWR68xx).

3.2.5 profileCfg command

An other point is the profileCfg command. This one allows to configure the chirp waveforms. As shown in the figure 2, different parameters are useful for the FMCW chirp signal. All of these have to be given in the profileCfg command. The description of this command is shown below :

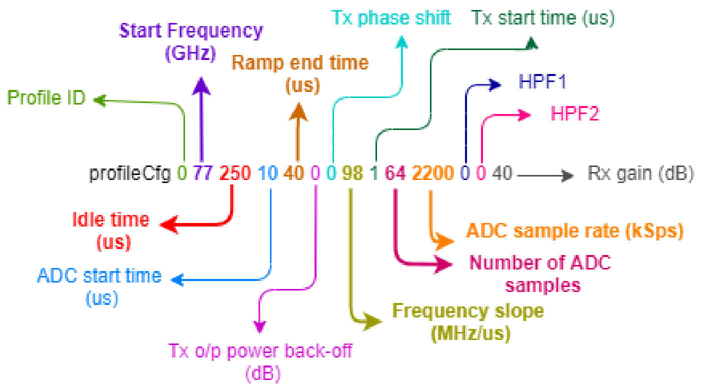


FIGURE 14 – profileCfg configuration.

— **profile ID** is the profile identifier. Depends on the dfeOutputMode value, two mode are possible :
  — Legacy frame (dfeOutputMode = 1) : just a single valid profile per configuration is supported.

  — Advanced rfame (dfeOutputMode = 3) : It is possible in the radar's **advanced mode** to have multiple chirp configurations in a single frame by dividing the frame into sub-frames. One profile per subframe is supported. However, different subframes can have different profiles.

— **Start Frequency** is the Frequency Start in GHz for the beginning of the total occupied bandwidth (refers to figure 2). With the **Frequency slope** and the **ADC Start Time**, they configure the valid occupied bandwidth. Increasing the frequency slope improves the range resolution and decreases the maximum unambiguous range.

— **Idle time** is the idle time between two consecutive chirps in usec. With the **ramp end time**, they configure the velocity sensitivity. If the developer wants to modify motion sensitivity, he has to change those parameters by ensuring the total chirping time and the required processing time fit into a single frame period. Furthermore, for the **ramp end time**, we have :

$$\text{ramp end time} = \text{ADC Start Time} + \text{ADC sampling time} + \text{excess time}$$

More precisely, those parameters affect the inter-chirp time. Thus, velocity resolution and maximum unambiguous velocity. Assuming that the number of chirps per frame is constant, increasing the inter-chirp time improves the velocity. However, it decreases the maximum unambiguous velocity. Be careful, increase the inter-chirp time logically increase the total chirping window but decrease the total available processing time within the frame period. By this way, take care about the value of frame Periodicity in frameCfg command to ensure that the total chirping time and the required processing time can fit into a frame period.

— **TX output power back-off** is the concatenation of the code of output power back off for TX1, TX2 and TX3. The bit description of the TX output power back off code is :
— b7 : 0 TX1 o/p power back off in dB scale
— b15 : 8 TX2 o/p power back off in dB scale
— b23 : 16 TX3 o/p power back off in dB scale
— b31 : 24 Reserved
Noted that only "0" has been tested for the provided demos. This parameter affects on the maximum detection range and need to be configured by limiting yourself to the desired requirements.

— **txPhaseShifter** is the tx phase shifter for the tx antennas. Noted that only "0" has been tested for the provided demos.

— **Number of ADC samples** is those which are collected during the "ADC sampling time". This parameter has an impact on the bandwidth and the chirp time. Assuming that the frequency slope and the sampling rate are constant, it manages the range resolution and the valid chirping time and maximum detectable range based on SNR. By increasing this parameter, bandwidth will be larger. Hence, range resolution will be improved. Also, the number of ADC samples impacts on the radar cube size but need processing power. Be careful as the result can be running out of memory.

— **ADC sample rate** is the ADC sampling frequency in kSps. It is also called "digOut-SampleRate". This parameter impacts on the chirp time and bandwidth. Assuming that the frequency slope and the number of ADC samples are constant, it manages the range resolution and the maximum unambiguous range. Thus, the bandwidth will be shorter, the range resolution will be worst but the maximum unambiguous range will be better.

— **HPF1** is the High Pass Filter 1 corner frequency :
— 0 : 175 kHz
— 1 : 235 kHz
— 2 : 350 kHz

  - 3 : 700 kHz
- **HPF2** is the High Pass Filter 2 corner frequency :
  - 0 : 350kHz
  - 1 : 700 kHz
  - 2 : 1.4 MHz
  - 3 : 2.8 MHz

3.2.6 chirpCfg command

The next point is the chirpCfg command. This one allows to configure the TX antennas. This command have to be run three times consecutively per TX antenna. The description of this command is shown below :

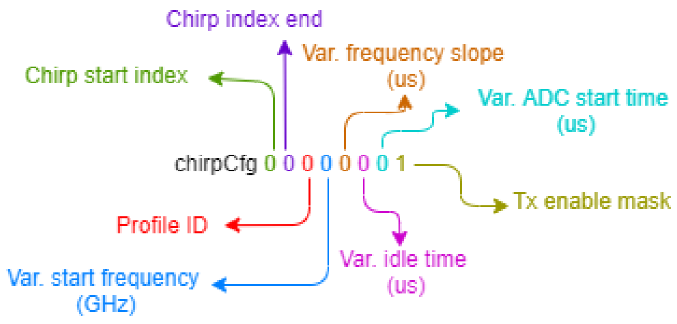


FIGURE 15 – chirpCfg configuration.

- **Chirp Start index** is as its name indicates, the chirp start index. Depends of the Tx antenna, it is can be set to 0 (for TX1), 1 (for TX2) or 2 (for TX3).Possible values : 0-511
- **Chirp End index** is as its name indicates, the chirp end index. It is set in the same way as the **Chirp Start index**. Possible values :0-511
- **Profile ID** is the profile identifier. It has to match with the profilCfg-> profilId. Possible values :0-3
- **Var Start Frequency** is the Start frequency variation in Hz. Only "0" has been tested with the demos. Configure the start frequency in the **profileCfg**.
- **Var Frequency Slope** is the frequency slope variation in kHz/us. Only "0" has been tested with the demos. Configure the frequency slope in the **profileCfg**.
- **Var Idle Time** is the Idle time variation in us. Only "0" has been tested with the demos. Configure the idle time in the **profileCfg**.
- **Var ADC Start time** is the ADC start time variation in us. Only "0" has been tested with the demos. Configure the ADC start time in the **profileCfg**.
- **TX enable mask** is as its name indicates the TX antenna mask. Only one distinct TX antenna can be enabled per chirp. Depends on what TX desired, set it to 1 (for TX1), to 2 (for TX2) or to 4 (for TX3).

3.2.7 frameCfg command

An other point is the frameCfg command. This one configures the number of chirp loops in a single frame. Noted that dfeOutputMode need to be set to 1 to use this command. The description of frameCfg command is shown below :

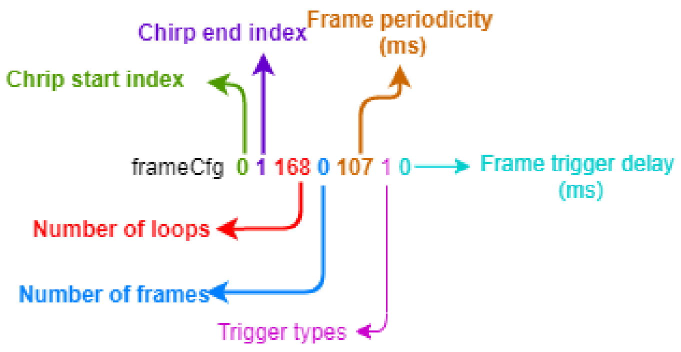


FIGURE 16 – frameCfg configuration.

— **Chirp Start index** is as its name indicates, the chirp start index. Any value are possible but chirpCfg command need to correspond.

— **Chirp End index** is as its name indicates, the chirp end index. Any value are possible but chirpCfg command need to correspond.

— **Number of loops** is the number of loops in a single frame. By this way, it is the total number of chirps transmitted for each TX in a single frame. It affects the velocity resolution and the maximum detection range based on SNR. Indeed, increasing this parameter improve the velocity resolution. However, it reduces the total available processing time within a frame period.

— **Number of frames** is at its name indicates the number of total frames. Under the assumption that it continuous mode is required, set it to 0.

— **Frame periodicity** is at its name indicates the frame periodicity in ms. Take care to fit the total chirping time and the required processing time in a single frame period.

— **Trigger types** is at its name indicates the trigger select :
  — 1 : software trigger.
  — 2 : hardware trigger.
  Noted that just Software trigger is supported. Thus, set it to 1.

— **Frame trigger delay** is at its name indicates the frame trigger delay in ms. Set it to 0.

3.2.8 cfarCfg command

An other command in the configuration file is the cfarCfg one wichi allows to configure the CFAR. It is constructed as below :

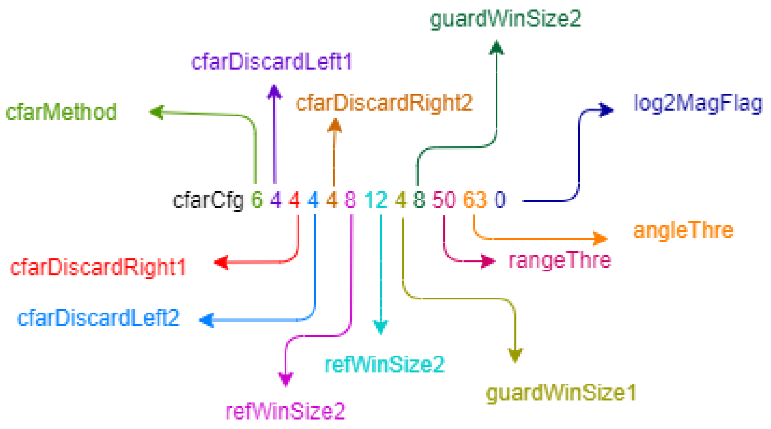


FIGURE 17 – cfarCfg configuration.

— **cfarMethod** corresponds as its name indicates to the CFAR method. In the case of the valu is 6, it corresponds to 2-pass range-azimuth CFAR.

— **cfarDiscardLeft1** corresponds to samples discarded on the left in terms of range.

— **cfarDiscardRight1** corresponds to samples discarded on the right in terms of range.

— **cfarDiscradLeft2** corresponds to samples discarded on the left in terms of angle.

— **cfarDiscradRight2** corresponds to samples discarded on the right in terms of angle.

— **refWinSize1** is the range reference win size.

— **refWinSize2** is the angle reference win size.

— **guardWinSize1** is the range reference guard size.

— **guardWinSize2** is the angle reference guard size.

— **rangeThre** is the range threshold * 10.

— **angleThre** is the angle threshold * 10.

— **log2MagFlag** corresponds as its name indicates to the log2 Magnitude Flag.

All the other configuration parameters used are outlined in Appendix A.4

3.3 Influence of chirp configuration parameters on the system's performances

3.3.1 Default configuration file for sense and direct 3d people counting demo

For the sense and direct people counting demo on industrial toolbox (in people counting demo), the default configuration file is as follows :

dfeDataOutputMode
dfeDataOutputMode
1

channelCfg		
rxChannelEn	txChannelEn	cascading
15	5	0
4 Rx	Tx1 and Tx3	no cascading

adcCfg	
numADCBits	adcOutputFmt
2	1

adcbufCfg			
adcOutputFmt	SampleSwap	chanInterleave	chirpThreshold
0	1	1	1

profileCfg			
profileId	startFreq (Ghz)	idleTime (us)	adcStartTime (us)
0	62.00	30	10
rampEndTime (us)	txOutPower	txPhaseShifter	freqSlopeConst(Mhz/us)
69.72	0	0	28.42
txStartTime (us)	numAdcSamples	digOutSampleRate (Ksps)	hpfCornerFreq1
1	128	2180	0
hpfCornerFreq2	rxGain (dB)		
0	24		

chirpcfg (Tx1)			
startIdx	endIdx	profileId	Var.startFreq (Hz/usec)
0	0	0	0
Var.freqSlopeVar (Hz/usec)	Var.idleTime (us)	Var.adcStartTime (us)	txEnable
0	0	0	1

chirpcfg (Tx3)			
startIdx	endIdx	profileId	Var.startFreq (Hz/usec)
1	1	0	0
Var.freqSlopeVar (Hz/usec)	Var.idleTime (us)	Var.adcStartTime (us)	txEnable
0	0	0	4

frameCfg			
chirpStartIdx	ChirpEndIdx	NumLoops	NumFrames
0	1	128	0
framePeriodicity (ms)	triggerSelect	frameTriggerDelay (us)	
50	1	0	

lowPower	
<don't care>	ADC mode
0	0

guiMonitor			
detected objects	tracker output	classification output	reducePointCloudOutputSize
1	1	1	1

cfarCfg			
cfarMethod	cfarDiscardLeft1 (samples)	cfarDiscardRight1 (samples)	cfarDiscardLeft2 (samples)
6	4	4	4
cfarDiscardRight2 (samples)	refWinSize1	refWinSize2	guardWinSize1
4	8	12	4
guardWinSize2	rangeThre	angleThre	log2MagFlag
8	50	63	0

doaCfg		
doaSearchRange	doaSearchRes	gamma
600	666	30
clutterRemovalOn	doaDopplerOversamplingFactor	doaDopplerSearchUsingCFAR
1	1	1
doaDopplerSearchCFARThr	doaDopplerSearchCFARGuard	doaDopplerSearchSNRReport
300	4	2

adcbufCfg			
adcOutputFmt	SampleSwap	ChanInterleave	chirpThreshold
0	1	1	1

gatingParam			
gating volume	length limit (m)	width limit (m)	velocity limit (m/s)
3	1.5	1.5	0

StateParam		
det2actThre	det2freeThre	active2freeThre
3	3	10
static2freeThre	exit2freeThre	sleep2freeThre
40	5	600

allocationParam					
snrThre	snrThreobscured	velocityThre	pointsThre	maxDistanceThre	maxVelThre
300	800	0.1	30	0.5	20

SceneryParam			
leftX (m)	leftY (m)	nearY (m)	farY (m)
-5	5	0.25	10

FilterParam		
HumanDisplacement	ifUniqueness	ifDistance
2.0	0.5	1.5

trackingCfg			
enableFlag	targetTyp	maxMeasPnt	maxTracks
1	2	300	15
maxRadialVel	velResolution	frameRate (ms)	sensorBoreSightAnglehorizontal (degrees)
67	105	50	90

classifierCfg			
enableFlag	classifierType	k	codebookSize
1	1	3	500
gamma	neighborDistSqrThr	gamma1	minNpntsPobj
0.8	1.0	0.95	10

Key parameters	
Parameters	Values
Bandwidth (MHz)	1670
idle time (us)	30
adcStartTime (us)	10
frequency slope (Mhz/us)	28.42
num. chirps per frame	256
Frame duration (ms)	50
Sampling rate (ksps)	2180
adcSamplingTime (us)	128/2180= 58,7
AdcSampling time + Adc Valid Start Time + 1us	69.7 us
rampEndTime (us)	69.72
Total chirp time (us)	199
Maximum range, Rmax (m)	10,35
Range resolution (m)	0,0898 (8.98 cm)
Maximum velocity (m/s)	6,065
Velocity resolution (m/s)	0,0473

* Ksps = Kilo samples per second

* The range resolution is 8.98 cm so only points that are 8.98 cm apart in range, or separated in velocity at the same range bin are detected.

3.3.2 Computations

- Speed of light **C** = 3e8

- TotalchirpTime, **Tc** (us) = (idleTime(us) + rampEndTime(us)) * number of chirps

- **lda** = C / startFreq(GHz) * 1e9)

$$\text{Bandwidth (Ghz)} = \frac{numAdcsamples}{AdcSamplingRate(ksps)} * Freq.Slope(Mhz/us)$$

$$\text{maxRange} = \frac{AdcSamplingRate(ksps) * 0.9 * C}{2 * freq.Slope(Mhz/us) * 1e12}$$

$$\text{Range Resolution} = \frac{C}{(2 * Bandwidth)}$$

$$\text{max Velocity} = \frac{lda}{(4 * Tc(s))}$$

$$\text{velocity Resolution} = \frac{lda}{(2 * Tc(s) * numLoops * numTxantennas)}$$

3.3.3 Configuration restrictions

Along with restrictions specific to each chirp parameters, there are other restrictions to keep in mind when modifying the configuration parameters. 3.2.

— Startfrequency should be between 57GHz to 64GHz for the AWR6843. The maximum bandwidth is 4000Mhz

— $AdcSamplingTime = \frac{numAdcSamples}{digOutSampleRate}$

— Adc Sampling time + Adc Valid Start Time (adcStartTime) + 1us (minimum margin) <= rampEndTime.

— Total chirp time [(ramp end time + idle time)* number of chirps (# of chirpsCfg)] should be at least 50% lower than the Frame periodicity.

— The total chirping time and the required processing time should fit into a single frame period

— Number of loops should have a value between 1 and 255. For xWR68xx demos, as it is the DSP version of Doppler DPU used, the number of loops need to me a multiple of 4.

— The AWR6843 radar module has a L3 radar data cube RAM of 768 KB. The radar data cube RAM stores the data arising from the ADC samples corresponding to a frame.
The required radar memory (in KB (Kilobyte)) can be calculated as = (#RxAntennas * #TxAntennas * #rangefft-bins * var * sampleSize)/ 8 / 1024

Where var is either the number of chirp loops when data is written back to the radar cube or the number of doppler bins where there is no write back. By default var = number of chirp loops

And sampleSize is equal to 2*16 = 32 bit for complex sampling . 2 for I and Q signals

3.4 modification of configuration parameters

The aim of this part is to determine the influence of some parameters on the system. One single parameter from the default configuration has been changed per modification (modif.) to highlight its impact. However, for real tests on the AWR6843 radar sensor, some other parameters will need to be modified as well in order to respect the given restrictions.

3.4.1 Modification of the Number of Loops : modif. 1, 2

The first modification done is the number of loops in frameCfg command as below :

frameCfg	
modified parameter	NumLoops
modified value (1st modif.)	128 $\longrightarrow$ 16
modified value (2nd modif.)	128 $\longrightarrow$ 252

Number of chirps per frame :
1st modif. : (end chirp index - start chirp index + 1)*numloop = 2*16 = 32
2nd modif. : 2*252 = 504

3.4.2 Modification of the Number Tx antennas : modif. 3

The next modified parameter is the number Tx antennas. For that, there is different modifications to do :

Modify the txChannelEn in channelCfg and ChirpEndIdx in frameCfg command to add a Tx antenna :

channelCfg	
modified parameter	txChannelEn
modified value (modif. 3)	5 $\longrightarrow$ 7

frameCfg	
modified parameter	ChirpEndIdx
modified value (modif. 3)	1 $\longrightarrow$ 2

Furthermore, configure the TX2 antenna and reconfigure TX3 (the first chipCfg remains unchanged) :

chirpcfg (Tx2)			
startIdx	endIdx	profileId	startFreq (Hz/usec)
1	1	0	0
freqSlopeVar (Hz/usec)	idleTime (us)	adcStartTime (us)	txEnable
0	0	0	2

chirpcfg (Tx3)			
startIdx	endIdx	profileId	startFreq (Hz/usec)
2	2	0	0
freqSlopeVar (Hz/usec)	idleTime (us)	adcStartTime (us)	txEnable
0	0	0	4

Furthermore, configure the TX2 antenna (**bis**) the first chipCfg (TX1) and second chirpCfg (TX3) remain unchanged) :

chirpcfg (Tx2)			
startIdx	endIdx	profileId	startFreq (Hz/usec)
2	2	0	0
freqSlopeVar (Hz/usec)	idleTime (us)	adcStartTime (us)	txEnable
0	0	0	2

3.4.3 Modification of profile parameters

3.4.3.1 StartFrequency : modif. 4,5

profileCfg	
modified parameter	startFrequency
modified value (modif. 4)	62 $\longrightarrow$ 60
modified value (modif. 5)	62 $\longrightarrow$ 64

3.4.3.2 numAdcSamples : modif. 6, 7

profileCfg	
modified parameter	numAdcSamples
modified value (modif. 6)	128 $\longrightarrow$ 96
modified value (modif. 7)	128 $\longrightarrow$ 252

For the modif. 6 it comes : ADC sampling time = numAdcSamples / digOutSampleRate = 96 (samples) / 2180 (ksps) = 0.096/2180 = **44.04 us**.

For the modif. 7 it comes : ADC sampling time = numAdcSamples / digOutSampleRate = 252 (samples) / 2180 (ksps) = 0.252/2180 = **115.60 us**.

Furthermore, we have :

$$\text{ramp end time} = \text{ADC Start Time} + \text{ADC sampling time} + \text{excess time}$$

with 1 us (minimum) of excess time, it comes :

For the modif. 6 : ramp end time = 44.04 + 10 + 1 = **55.04 us**.

For the modif. 7 : ramp end time = 115.60 + 10 + 1 = **126.60 us**. Hence :

profileCfg	
modified parameter	rampEndTime (us)
modified value (modif. 6)	69.72 $\longrightarrow$ 55.04
modified value (modif. 7)	69.72 $\longrightarrow$ 126.60

Considering the restriction about the fact that active chirp time (rampEndTime + idleTime) should be at least 50% lower than the Frame periodicity, and idleTime is 30 us (profilCfg -> idleTime), it comes :

For modif. #6 :

$$\text{chirp time} = \text{rampEndTime} + \text{idleTime} = 55.04 + 30 = \mathbf{85.04\ us}$$

$$\text{chirp repetition time*} = (\text{ chirp time})\text{* num of chirps}(\#\text{ChirpCfg}) = 85.04 * 2 = \mathbf{170\ us}$$

*see figure **??**

For modif. #7 :

$$\text{chirp time} = \text{rampEndTime} + \text{idleTime} = 126.60 + 30 = \mathbf{156.60\ us}$$

$$\text{chirp repetition time*} = 156.6 * 2 = \mathbf{313\ us}$$

*see figure **??**

As the frequency period is 50ms or 50 000us, this restriction is verified for both modif.s.

3.4.3.3 digOutSampleRate : modif. 8, 9

profileCfg	
modified parameter	digOutSampleRate
modified value (modif. 8)	2180 $\longrightarrow$ 300
modified value (modif. 9)	2180 $\longrightarrow$ 3000

Max range = 1.4 m.

Therefore, SceneryParam FarY parameter is changed from 10 $\longrightarrow$ 1.4

3.4.3.4 idleTime : modif. 10, 11

profileCfg	
modified parameter	idleTime
modified value (modif. 10)	30 $\longrightarrow$ 2
modified value (modif. 11)	30 $\longrightarrow$ 200

3.4.3.5 frequency slope : modif. 12

profileCfg	
modified parameter	freqSlopeConst(MHz/us)
modified value (modif. 12)	28.42 $\longrightarrow$ 70

Max range = 4.2 m

Therefore, SceneryParam FarY parameter should be changed from 10 $\longrightarrow$ 4

3.4.4 Comparison tables

Note : if a parameter is not mentioned in the tables, it means its value didn't change compared to default. For default values of all parameters, see 3.3.1

adcstarttime= 10 us and frame duration = 50 ms for the modif.s.

Key parameters	Values					
Parameters	default	modif. 1	modif. 2	modif. 3	modif. 4	modif. 5
changed parameter		num. loops	num loops	+1 TX	start freq.	start freq.
Start freq.(Ghz)	62	62	62	62	60	64
Bandwidth(Mhz)	1670	1670	1670	1670	1670	1670
idle time(Ghz)	30	30	30	30	30	30
num loops	128	16	252	128	128	128
num. chirps per frame	256	32	504	256	256	256
num.TX antennas	2	2	2	3	2	2
Total chirp time (us)	199	199	199	299	199	199
Maximum range, Rmax (m)	10,35	10,35	10,35	10,35	10,35	10,35
Range resolution (m)	0,0898	0,0898	0,0898	0,0898	0,0898	0,0898
Maximum velocity (m/s)	6,065	6,065	6,065	4.043	6.267	5,8758
Velocity resolution (m/s)	0,0473	0.379	0.024	0.021	0.0489	0,04590

Key parameters	Values							
Parameters	default	mod. 6	mod.7	mod.8	mod.9	mod.10	mod.11	mod.12
changed parameter		numAdcSamples		SampleRate		idleTime		F.Slope
Bandwidth(Mhz)	1670	1250	3290	1210	1210	1670	1670	4110
idle time(Ghz)	30	30	30	30	30	2	200	30
rampEndTime (us)	69.72	55.04	126.6	69.72	69.72	69.72	69.72	69.72
Freq.Slope (Mhz/us)	28.42	28.42	28.42	28.42	28.42	28.42	28.42	70
num.Samples	128	96	252	128	128	128	128	128
Adc Sampling Rate (ksps)	2180	2180	2180	300	3000	2180	2180	2180
Adc Sampling Time (us)	58.7	44,03	115.6	58.7	58.7	58.7	58.7	58.7
Total chirp time (us)	199	170	313	199	199	143	539	199
Maximum range, Rmax (m)	10,35	10,35	10,35	1.425	14,25	10.35	10.35	4.204
Range resolution (m)	0,0898	0,12	0,0456	0,012	0,1237	0,0898	0,0898	0.0364
Maximum velocity (m/s)	6,065	7,112	3,862	6,065	6,065	8.433	2.242	6,065
Velocity resolution (m/s)	0,0473	0.0555	0.0301	0,0473	0,0473	0,0658	0.0175	0.0473

Refer to **??** for calculation details.

Comments :

1- The more chirps there is in a frame, the better the velocity resolution.

2- We can improve the velocity resolution by having more Tx antennas or by increasing the start frequency or the number of adc samples per chirp or the idle time or the inter-chirp time.

However, when the radar becomes more sensitive to motion (better velocity resolution), its maximum velocity decreases.

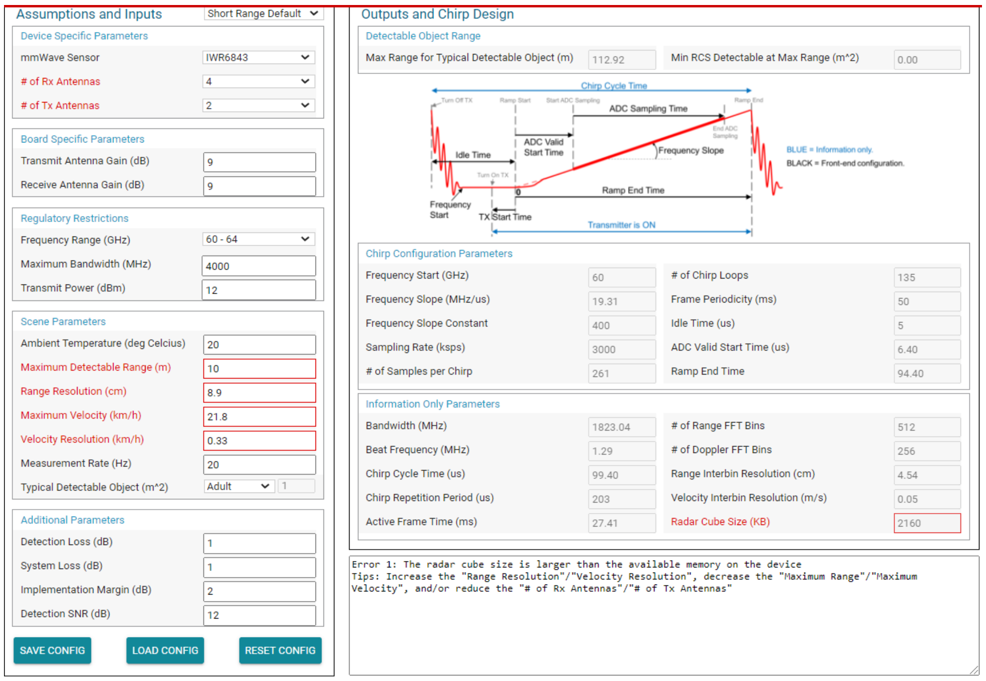
3- The Range and velocity resolutions can be improved by increasing the number of adc samples per chirp. But then again the maximum velocity will decrease.

4-By increasing the sampling rate, we can improve the maximum range of our application. However, the range resolution will suffer from the increase in maximum range.

While, increasing the frequency slope improves the range resolution and decreases the maximum unambiguous range

As we stated, only one parameter as-well as its **intrinsically** linked parameters has been changed per configuration file modification to determine influence of some parameters on the system. However, not all radar's restrictions were respected (i.e maximum radar cube size). For instance, **modif. 1** configuration file is respectful of the restrictions while **modif. 2** is not.

We highly recommend using TI's *mmwave sensing Estimator* when modifying the configuration parameters as the software can indicate whether or not the restrictions are respected. In the example below, the wanted radar performances (Scene parameters) require a radar cube size of 1620 Kb which is more than the max radar cube size of the AWR6843.



Device Specific Parameters

mmWave Sensor	IWR6843
# of Rx Antennas	4
# of Tx Antennas	2

Board Specific Parameters

Transmit Antenna Gain (dB)	9
Receive Antenna Gain (dB)	9

Regulatory Restrictions

Frequency Range (GHz)	60 - 64
Maximum Bandwidth (MHz)	4000
Transmit Power (dBm)	12

Scene Parameters

Ambient Temperature (deg Celcius)	20
Maximum Detectable Range (m)	10
Range Resolution (cm)	8.9
Maximum Velocity (km/h)	21.8
Velocity Resolution (km/h)	0.33
Measurement Rate (Hz)	20
Typical Detectable Object (m^2)	Adult 1

Additional Parameters

Detection Loss (dB)	1
System Loss (dB)	1
Implementation Margin (dB)	2
Detection SNR (dB)	12

Detectable Object Range

Max Range for Typical Detectable Object (m)	112.92	Min RCS Detectable at Max Range (m^2)	0.00

Chirp Configuration Parameters

Frequency Start (GHz)	60	# of Chirp Loops	135
Frequency Slope (MHz/us)	19.31	Frame Periodicity (ms)	50
Frequency Slope Constant	400	Idle Time (us)	5
Sampling Rate (ksps)	3000	ADC Valid Start Time (us)	6.40
# of Samples per Chirp	261	Ramp End Time	94.40

Information Only Parameters

Bandwidth (MHz)	1823.04	# of Range FFT Bins	512
Beat Frequency (MHz)	1.29	# of Doppler FFT Bins	256
Chirp Cycle Time (us)	99.40	Range Interbin Resolution (cm)	4.54
Chirp Repetition Period (us)	203	Velocity Interbin Resolution (m/s)	0.05
Active Frame Time (ms)	27.41	Radar Cube Size (KB)	2160

```
Error 1: The radar cube size is larger than the available memory on the device
Tips: Increase the "Range Resolution"/"Velocity Resolution", decrease the "Maximum Range"/"Maximum
Velocity", and/or reduce the "# of Rx Antennas"/"# of Tx Antennas"
```

Figure 18 – Required radar cube size

4 Bibliography

[1] https://www.ti.com/lit/an/swra553a/swra553a.pdf?ts=1624345333114&ref_url=https%253A%252F%252Fwww.google.com%252F

[2] https://dev.ti.com/tirex/explore/node?node=AIQzhJRYjeBjkiRvsK8EVg__VLyFKFf__LATEST

[3] https://dev.ti.com/tirex/explore/content/mmwave_industrial_toolbox_4_8_0/labs/people_counting/docs/3D_people_counting_demo_implementation_guide.pdf

[4] https://dev.ti.com/tirex/explore/content/mmwave_industrial_toolbox_4_8_0/labs/people_counting/docs/3D_people_counting_detection_layer_tuning_guide.pdf

[5] Radar Tutorial : https://www.radartutorial.eu/02.basics/Frequency%20Modulated%20Continuous%20Wave%20Radar.en.html

[6] https://www.researchgate.net/publication/329501527_On_fundamental_operating_principles_and_range-doppler_estimation_in_monolithic_frequency-modulated_continuous-wave_radar_sensors

[7] https://training.ti.com/sites/default/files/docs/mmwaveSensing-FMCW-offlineviewing_4.pdf

[8] https://www.ti.com/lit/wp/spyy007/spyy007.pdf

[9]http://software-dl.ti.com/ra-processors/esd/MMWAVE-SDK/latest/exports/mmwave_sdk_user_guide.pdf

[10] Radar tutorial https://www.radartutorial.eu/02.basics/Frequency%20Modulated%20Continuous%20Wave%20Radar.en.html

[11] https://www.researchgate.net/publication/337496208_Design_and_Implementation_of_A_Low_Cost_Fmcw_Radar_with_Configurable_Signal_Processor_for_Human_Movement_and_Breathing_Detection

[12] https://www.researchgate.net/publication/330951560_Assisting_the_visually_impaired_Multitarget_warning_through_millimeter_wave_radar_and_RGB-depth_sensors

[13] https://www.researchgate.net/publication/282360406_Wireless_non-invasive_continuous_respiratory_monitoring_with_FMCW_radar_a_clinical_validation_study

[14] https://publications.lib.chalmers.se/records/fulltext/255856/255856.pdf

[15] https://www.radartutorial.eu/02.basics/Radar%20%C3%A0%20onde%20continue%20modul%C3%A9e%20en%20fr%C3%A9quence.fr.html

[16] https://www.researchgate.net/publication/281299018_Radars_FMCW_pour_l%27aide_a_la_conduite_automobile

[17] https://oatao.univ-toulouse.fr/8608/1/goy.pdf

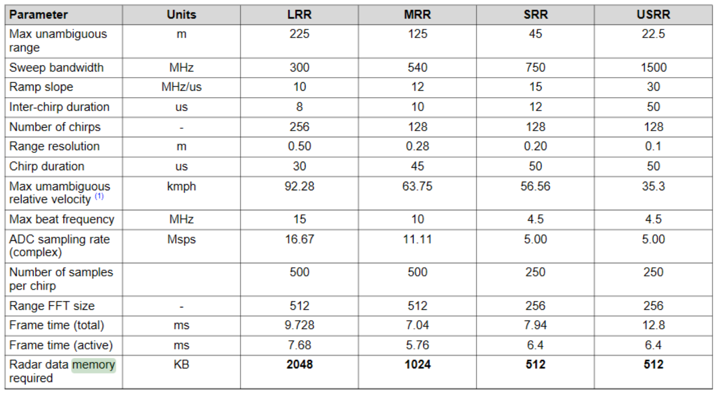
[18] https://publications.lib.chalmers.se/records/fulltext/255856/255856.pdf

[19] https://www.google.com/url?q=https://www.radartutorial.eu/02.basics/Radar%2520%25C3%25A0%2520onde%2520continue%2520modul%25C3%25A9e%2520en%2520fr%25C3%25A9quence.fr.html&sa=D&source=editors&ust=1624437254517000&usg=AOvVaw34qiSHwko4K3rJNjdTFLoM

A Appendix

A.1 Example of Chirp Configurations and how it affects the required data memory.

provided by TI [1]



Parameter	Units	LRR	MRR	SRR	USRR
Max unambiguous range	m	225	125	45	22.5
Sweep bandwidth	MHz	300	540	750	1500
Ramp slope	MHz/us	10	12	15	30
Inter-chirp duration	us	8	10	12	50
Number of chirps	-	256	128	128	128
Range resolution	m	0.50	0.28	0.20	0.1
Chirp duration	us	30	45	50	50
Max umambiguous relative velocity (1)	kmph	92.28	63.75	56.56	35.3
Max beat frequency	MHz	15	10	4.5	4.5
ADC sampling rate (complex)	Msps	16.67	11.11	5.00	5.00
Number of samples per chirp		500	500	250	250
Range FFT size	-	512	512	256	256
Frame time (total)	ms	9.728	7.04	7.94	12.8
Frame time (active)	ms	7.68	5.76	6.4	6.4
Radar data memory required	KB	**2048**	**1024**	**512**	**512**

A.2 Radar geometry

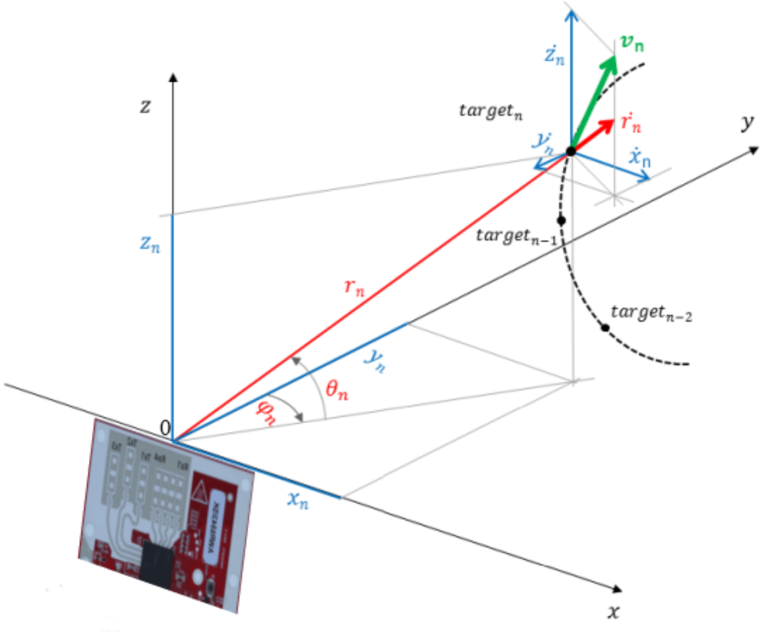


FIGURE 19 – radar geometry.

range (r), azimuth (φ), elevation (θ), and radial velocity ($\dot{r}$)

The cartesian coordinates of points in the point cloud are obtained from the polar coordinates.

For 3d : z = range *sin(doppler)
x= range * cos(doppler) *sin(azimuth)
y= = range * cos(doppler) * cos(azimuth)

For 2d (i.e sense and direct people counting demo) :
z = 0
x= range * sin(azimuth)
y= = range * cos(azimuth)

A.3 Group Tracker (Gtracker process)

The Group tracker or Gtracker module filters the detected points in a spatially and temporally way to localize and track clustered group of points.its process is given below :

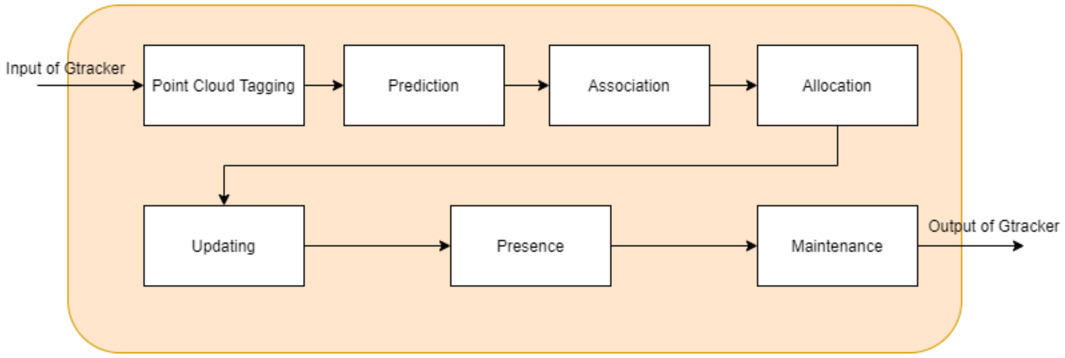


FIGURE 20 – GTracker process.

A.3.1 Point Cloud Tagging

This first step of the process is used to ignore targets which are outside of the defined zone. Indeed, depends on the desired area chosen by the developer, unwanted target can be to a certain extent ignored.

A.3.2 Prediction

This second step has as input the position, radial velocity and acceleration (i.e state vector) in Cartesian co-ordinates. Also it has the corresponding state covariance matrix of a tracked object. Thanks to the standard prediction equations of the Kalman Filter, it is possible to estimate the new position, radial velocity, acceleration (i.e a-priori state estimation) and corresponding state covariance matrix (i.e a-priori covariance estimation) for each target.

A.3.3 Association

This third step is used to associate each point with a single track. This association is done by using Mahalanobis distance between each measurement point and each of targets. To determine which track will be associated to each point, it is necessary to refer to the highest bidding score.

A.3.4 Allocation

This fourth step is used to treat the points which have not been associated with an existing track during the last step. Thus, this step is similar to create clusters. Indeed, a measurement point is arbitrarily chosen as leading one to initialize the cluster set. Then, all the non associated points are checked to verify if they can join this cluster (correct velocity and distance). Hence, cluster centroid is redefined. To count this cluster as a correct one, it has to include minimum number of points, a velocity higher than a velocity threshold and a strong enough combined SNR for the points inside.

A.3.5 Updating

This fifth step is used to update the track centroid and the corresponding covariance matrix estimates as new measurements are associated with a track. Furthermore, the group track dispersion matrix and group covariance matrix are also calculated in the updating step.

A.3.6 Presence

This sixth step is used to determine is there is any target in the occupancy area. For that, it is based on two indications :

— **raw detection** : it uses the candidate set created before by the allocation step. If the number of points and their mean velocity are higher than the corresponding thresholds, those points are still considered as present.

— **know target in the area** : it checks if the know target measurement centroid is stil in the occupancy area.

A.3.7 Maintenance

This last step is used to determine the state of the track. Indeed, a track can be in detect, active or free state by following a cycle of events. At this step, this state can be modified or a track can be de-allocated.

A.4 Other configuration file commands used

A.4.1 lowPower command

The lowPower command is the low power configuration :

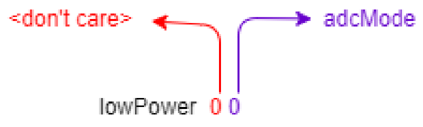


FIGURE 21 – lowPower configuration.

— The first parameter need to be set to 0 as is not used
— **adcMode** is the ADC Mode and can have two value :
  — 0x00 : Regular ADC mode
  — 0x01 : Low Power ADC mode (Not supported for xWR6xxx board).
  To determine the value, please check out the profilCfg -> digOutSampleRate

A.4.2 guiMonitor command

The guiMonitor command description is shown below :

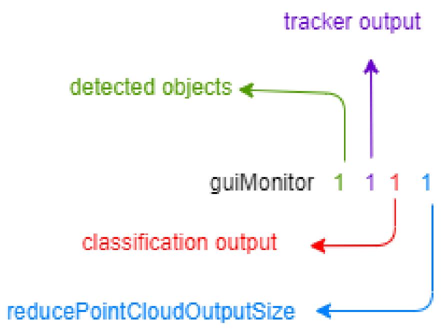


FIGURE 22 – guiMonitor configuration.

— **detected objects** enables or not the export of point cloud data :
  — 1 : enable export of point cloud (x,y,z,doppler) and point cloud sideinfo (SNR, noiseval).
  — 2 : enable export of point cloud (x,y,z,doppler).
  — 0 : disable

— **tracker output** sends or not target information from the tracker algorithm :
  — 1 : enable export of tracker output
  — 0 : disable

— **classification Output** allows export of object classification results with following tag Tag for targets if enabled : 1 :human, -1 :moving clutter :
  — 1 : enable export of classification Output.
  — 0 : disable

— **reducePointCloudOutputSize** reduce the point cloud output size to cope with UART limitation :
  — 1 :reduce of point cloud size.
  — 0 : no reduce

A.4.3 doaCfg command

The next command to study is the doaCfg. This command is useful to configure the distance of arrival (DOA). Indeed, the angular estimation can be further improved by using DSP techniques to estimate the DOA. Estimating it it possible thanks to the second axis of the radar cube which holds the spatially sampled signals using multiple virtual antennas. This command is configured as below :

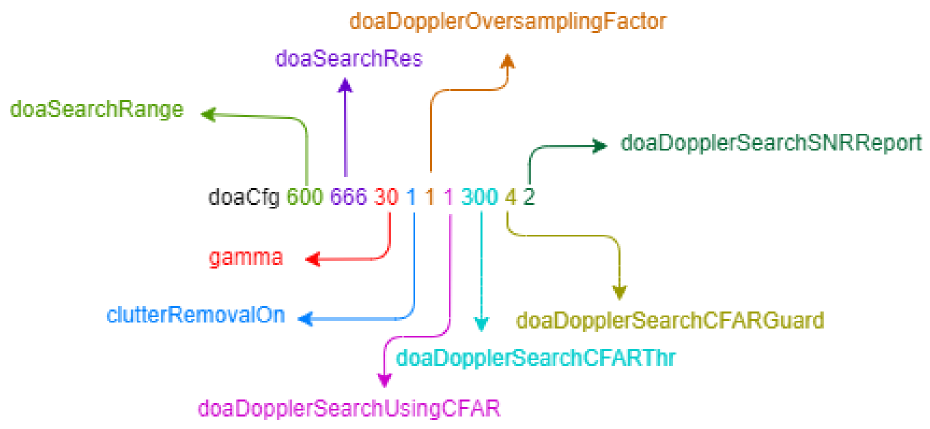

doaDopplerOversamplingFactor
doaSearchRes
doaSearchRange
doaDopplerSearchSNRReport
doaCfg 600 666 30 1 1 1 300 4 2
gamma
clutterRemovalOn
doaDopplerSearchCFARGuard
doaDopplerSearchCFARThr
doaDopplerSearchUsingCFAR

FIGURE 23 – doaCfg configuration.

— **doaSearchRange** corresponds to the DOA search Range and the displayed value is equal to 10*searchBound.

— **doaSearchRes** corresponds to the DOA Search Resolution and the displayed value is equal to 1000*searchRes.

— **gamma** corresponds to 1000*gamma.

— **clutterRemovalOn** corresponds to the Clutter Removal Flag.

— **doaDopplerOverSamplingFactor** is an integer oversampling factor.

— **doaDopplerSearchUsingCFAR** is a flag to indicate the use of CFAR for Doppler search.

— **doaDopplerSearchCFARThr** corresponds to 10*DopplerCFARThr.

— **DopplerCFARGuard** corresponds to the Doppler CFAR Guard Wine Size

— **doaDopplerSearchSNRReport** corresponds to the possibility to create a report as below :
  — 0 : report rangeDetSNR.
  — 1 : report scaled DopSNR.
  — 2 : reportDopSNR.

A.4.4 trackingCfg command

Furthermore, an other command is the trackingCfg one and configure the target and track measurement. It is constructed as below :

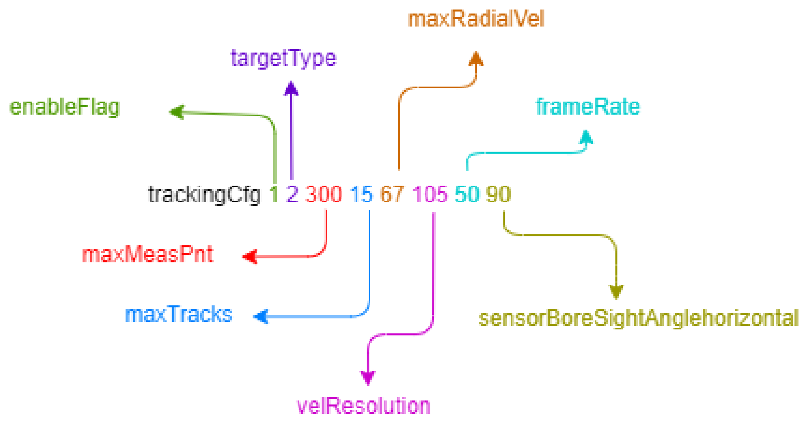


FIGURE 24 – trackingcfg configuration.

- **enableFlag** corresponds to the activation or not of the tracker :
  - 1 : enabled.
  - 0 : disabled.
- **targetType** configure the type of the target :
  - 1 : Vehicle.
  - 2 : People.
- **maxMeasPnt** corresponds to the maximum number of detection points per frame.
- **maxTracks** corresponds to the maximum number of targets to track.
- **maxRadialVel** corresponds to the maximum radial velocity. The displayed value is 10 * velocity.
- **velResolution** corresponds to the the radial velocity resolution in mm/sec.
- **frameRate** corresponds as its name indicates to the frame rate. The value displayed should match the sensor chirp configuration.
- **sensorBoreSightAngleHorizontal** corresponds to the angle in degrees between sensors' boresight and the horizontal axis (x-axis).

A.4.5 gatingParam command

For the tracker layer configuration, there are different parameters which have to be configured. The first one is the gating parameters thank to gatingParam command.

The gating parameters is used to set the boundary for each tracked unit for the Gtrack algorithm. It helps the algorithm track more efficiently the targeted objects by providing dimensional limits of objects that we wish to track. Hence, maximum volume and velocity are determined by this command.

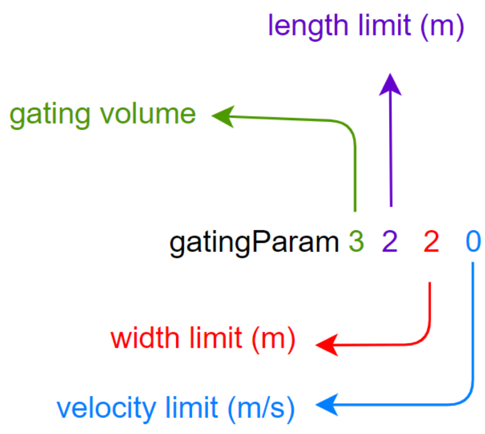


FIGURE 25 – gatingParam configuration.

— **gating volume or depth**
— **gating limit in length** is the maximum length of the target
— **gating limit in width** is the maximum width of target
— **gating limit in velocity** is the target's maximum velocity (with 0 meaning no limit)

A.4.6 stateParam command

This command corresponds to the second one to configure for the tracker layer configuration. It corresponds to the state parameters. This one is useful to determine the state of the tracking target : free, detect, or active.

— **Free** means that the track is not tracking anything.
— **Detect** means that the track has just been detected.
— **Active** means that the track has been around for some time.

Thus, when the allocation of a track is done, this one goes to **Detect** state. If the target is still detected after a while, it goes to **Active** state. In contrast, if this one disappear, it goes to **Free** state. Noted that in detect or active state, the track can be static or sleep tag. Static means that we think the track is still there but there is not any detected points whereas for sleep tag there are still detected points.

Hence, active state has three different threshold depends on the track tag : moving, sleep or static as show below :

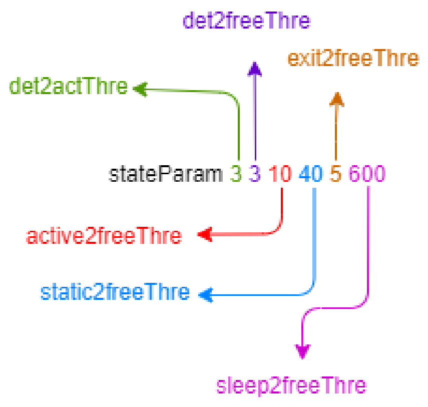


FIGURE 26 – stateParam configuration.

- **det2actThre** corresponds to the threshold for the number of continuous hits events (*in Detect state*) for the transition from detect to active. Hits events means that points are associated with a tracking instance.
- **det2freeThre** corresponds to the threshold for the number of continuous miss events (*in Detect state*) for the transition from detect to free. Miss events means that there are no points associated with a tracking instance.
- **active2freeThre** corresponds to the threshold for the number of continuous miss events for the transition from active to free.
- **static2freeThre** corresponds to the threshold for the number of continuous miss events for the transition from active to free. Static in the name corresponds to the static condition (static target into static zone based on the threshold value).
- **exit2freeThre** corresponds to the threshold for the number of continuous miss events when an active track is outside the static zone for the transition from active to free state.
- **sleep2freeThre** corresponds to the threshold for the maximum time target can be considered as static. Indeed, if a track has a sleep tag, a timer begins. If this one reach this threshold, the track will be dropped. However, if the track moves, the timer will be reset.

To illustrate these threshold, here is a scheme with the different transitions :

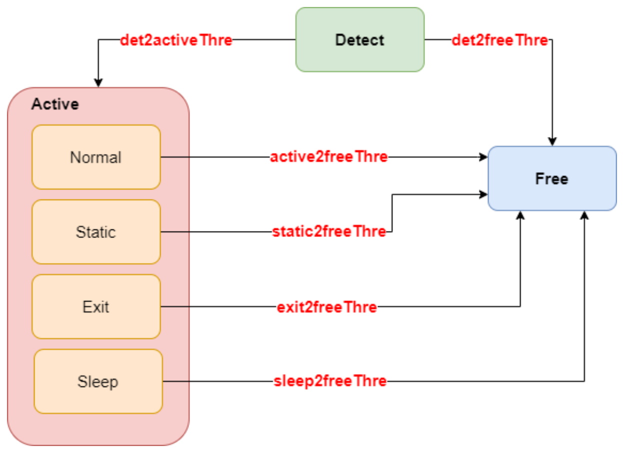


FIGURE 27 – Transitions for the stateParam command.

A.4.7 allocationParam command

The next command to be considered is the allocationParam. This one is for the track allocation. This parameter check if a candidate point can be bundled into an allocation set :

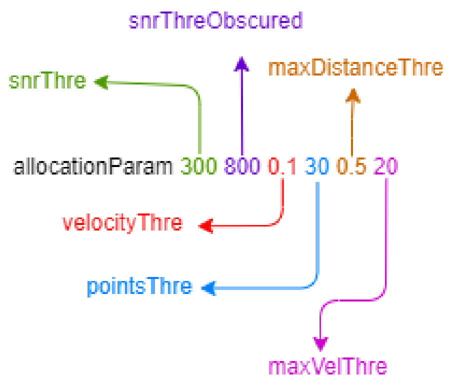


FIGURE 28 – allocationParam configuration.

- **snrThre** is the threshold for the minimum SNR for the allocation set. To define this value, it has to be about equal the expected linear SNR value for a single target at 6m multiplied by **pointsThre**.
- **snrThreObscured** is the threshold for the minimum SNR for the allocation set when the target is obscured by an other one. Furthermore, the target is described as obscured if the other target is behind and has a similar Doppler to an other existing target.
- **velocityThre** is the threshold for the minimum radial velocity of the allocation set center. Under this threshold, target will not be considered. Thus, to avoid undesirable reflections, this value needs to be a non-zero one but not so high to detect static target.
- **pointsThre** is the threshold for the minimum expected number of points from a target.
- **maxDistanceThre** is the threshold for the maximum squared distance from the center of allocation set and the potential candidate point. If the distance is under this threshold, this point will be clustered into the allocation set.
- **maxVelThre** is the the threshold for the maximum velocity difference between the center of allocation set and the potential candidate point. If the velocity difference is under this threshold, this point will be clustered into the allocation set. This threshold is useful to avoid Doppler error.

A.4.8 SceneryParam command

The next command is SceneryParam. This one is useful for the GUI and the Gtracker algorithm of Texas Instrument. It defines the dimensions of the space where the tracker will operate :

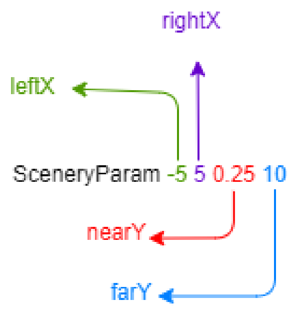


FIGURE 29 – SceneryParam configuration.

These parameters define the valid tracking area. People can only be tracked when they are inside the boxes. :

— **leftX** Left side of boundary box

— **rightX** right side of boundary box

— **nearY** close boundary parallel with the radar board

— **farY** Far boundary parallel with the radar

A.4.9 classifierCfg

the parameters help the detected objects classifier algorithm :

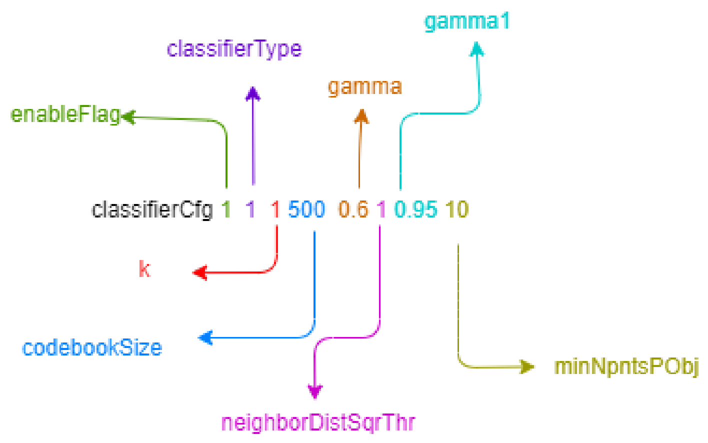


FIGURE 30 – classifierCfg configuration.

— **enableFlag** 1 enables classification ; 0 disables it

— **classifierType** 1 meaning K nearest neighbors classification

— **k** k value for k-nearest neighbors learning algorithm

— **codebookSize**

— **gamma** resistant factor : current output = previous output* gamma + current instantaneous classifier tag * (1-gamma)

— **neighborDistSqrThr** thereshold that sets the minimum square distance between 2 targets to declare close neighbor

— **gamma1** Close neighbor resistant factor for results smoothing. If a target has at least one close neighbor, gamma1 is used instead of gamma for smoothing. Note : gamma1 should be smaller than 1, but much bigger than 0.5, and bigger than gamma.

— **minNpntsPObj** minimum number of points in the target/track in the current frame to be taken into the statistics for classification

A.4.10 filterParam

This one puts a low pass filter on the object classifier. It's used in the source code gtrackfilter.c, mmwDemofiltercfg() :

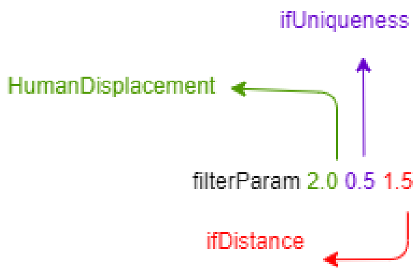


FIGURE 31 – filterParam.

Configuration information for TI's out of box demo can be found here [5]

4- Step by step Tutorial

A.5 Communication and Data structure : the TLV format

The radar sends and receives data through its serial ports. The user or enhanced port is used to send the configuration parameters to the board while tracking information is sent by the radar through its data or standard port encapsulated in frames.

Every frame consists of a fixed sized Frame Header followed by 1 or multiple number of TLVs depending on what was detected and on what information we want from the radar. The header contains important information such as the synchronisation pattern of magic word, the frame number and the number of TLVs in the frame.

The TLVs (Type-length-value structured data) can be of type Point Cloud, Target List, Target Index, etc. The type field helps therefore determine the type of data the current TLV item represents. The length field represents the length in bytes of each value and the value is the actual data which can be targets position or range and azimuth information depending on the the TLVs type. Please refer to [2] for further details on the datastructure.

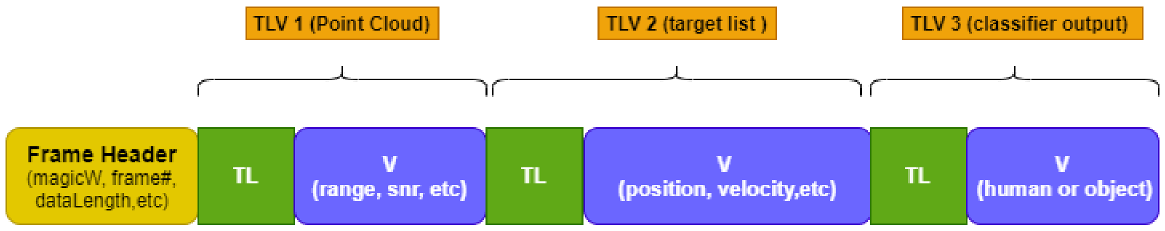


FIGURE 32 – TLV structure