

Vehicle Counting – Application Note – v2.x

Overview

Vehicle counting is an application that is totally solved with uRAD Industrial model + Tracking Software. It allows you to count vehicles in multiple lanes, measuring the velocity and classifying them, with high accuracy and minimal configuration. The system works at the 60 GHz band, an available frequency band for emission all over the world, which ease the certification of any product.

Use Case

The system is very versatile and can be used in many counting scenarios:

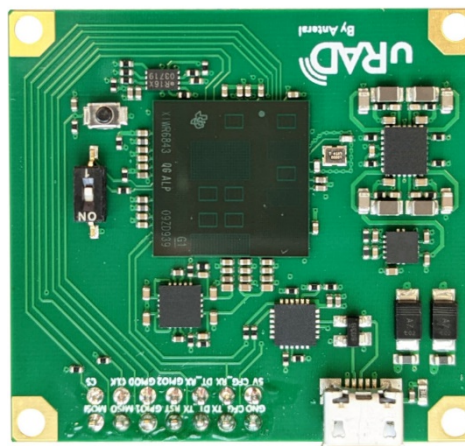
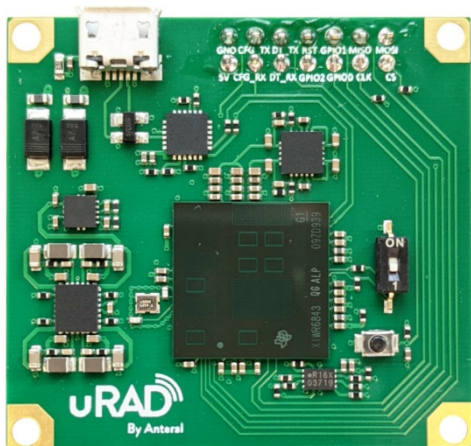
- Urban or interurban roads.
- Velocity measurement up to 180 km/h.
- Up to 6 lanes monitoring with a single radar.
- Counting vehicles with positive velocity (moving away) and negative velocity (approaching) at the same time.
- Dense or light traffic scenarios

Mounting

Three aspects have to be taken into account when mounting your vehicle counting system:

- **Radar orientation**

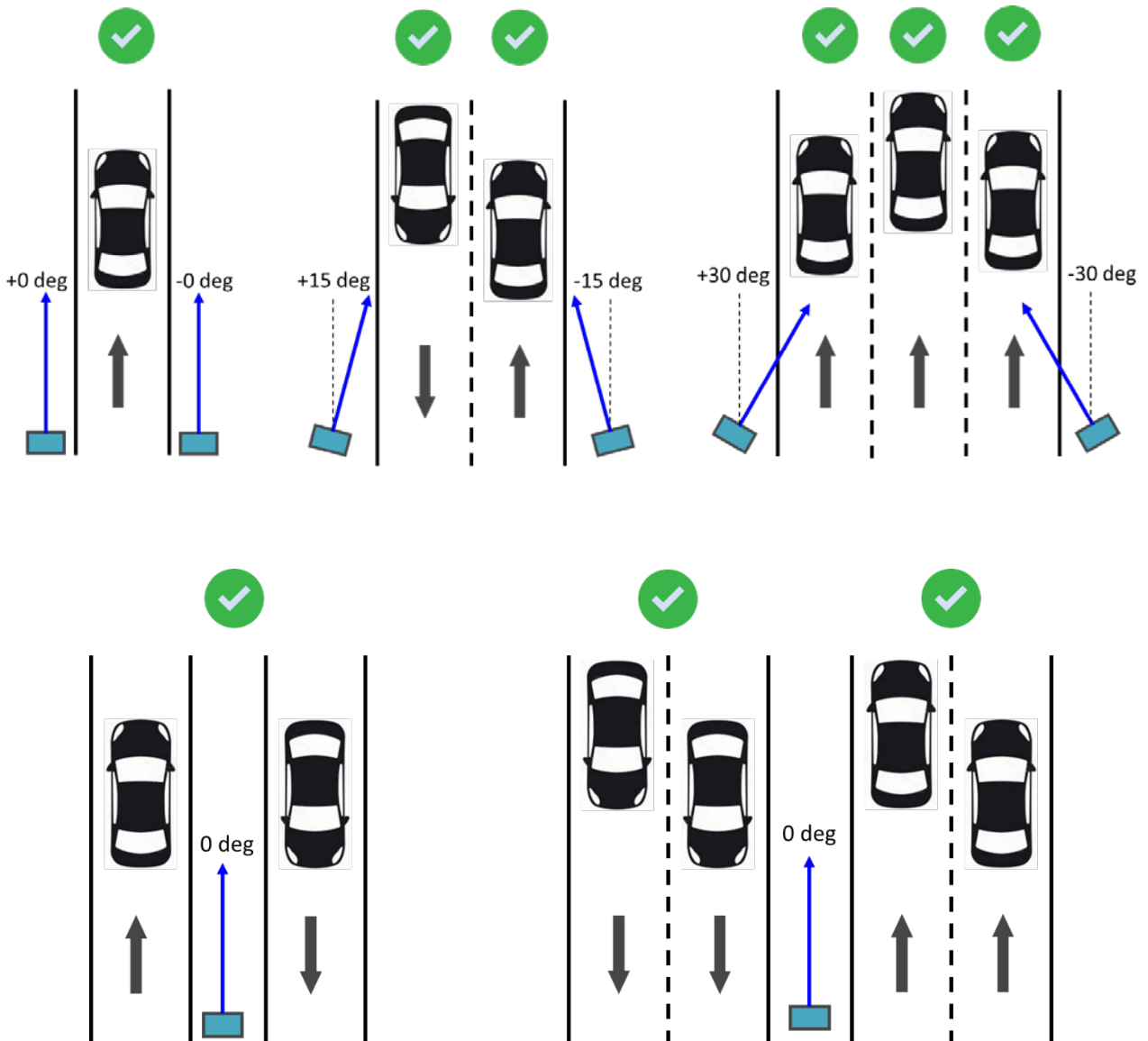
uRAD Industrial has to be mounted in the way that the USB connector is in the upper or lower side. Both options are correct.

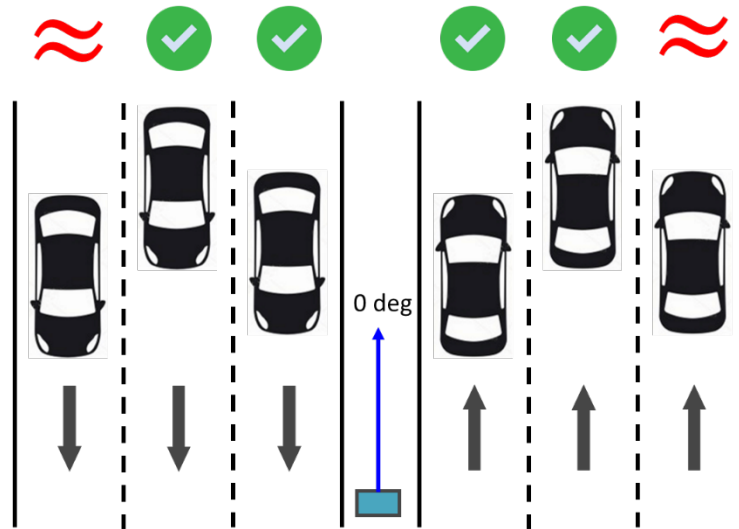


- **Radar installation in relation to the roadway**

uRAD Industrial can be mounted on the side of the road, either to the left or to the right, or above it. Depending on the use case, the general mounting recommendation is as follows: to measure **1 lane** on one side of the radar, mount it at a **0-degree** angle, for **2 lanes**, at a **15-degree** angle, and for **3 lanes**, at **30-degree** angle.

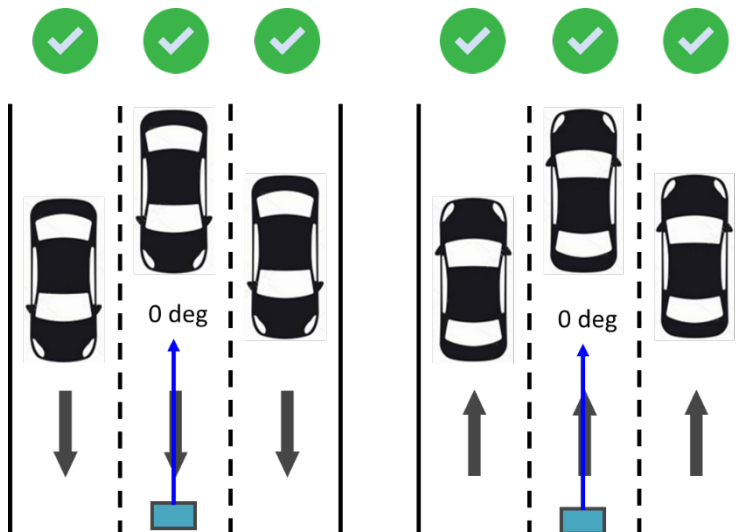
The following images illustrate the primary use cases and how the radar should be mounted.



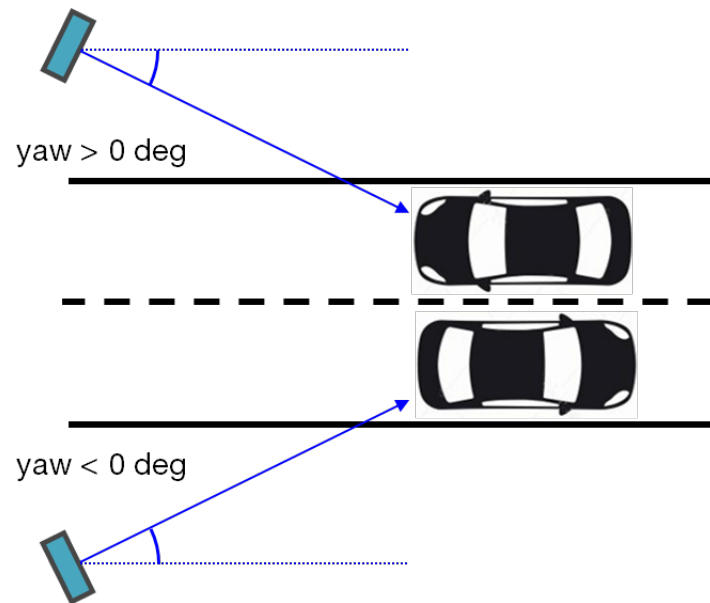


In the case of 6 lanes, the accuracy of counting in the outer lanes may be compromised depending on lane size and traffic density. **As a general rule, the device offers good accuracy up to 8 meters lateral distance.**

For this scenario, we recommend a radar for each direction.



In cases where it's necessary to provide a YAW angle, for example, in the case of 2 or 3 lanes or when an object obstructs the field of view, please **consider the sign of the angle** for configuration parameters according to the following image.



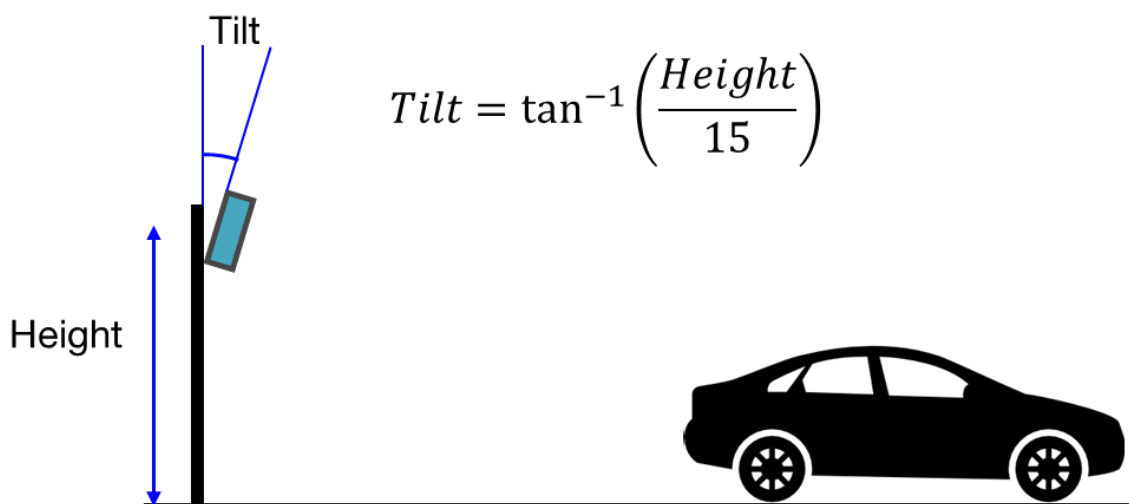
The radar is mounted **on the left side of the roadway**, and thus, cars pass by the right side of the radar (from an observer positioned behind the radar) → **YAW angle with a positive sign.**

The radar is mounted **on the right side of the roadway**, and thus, cars pass by the left side of the radar (from an observer positioned behind the radar) → **YAW angle with a negative sign.**

- **Radar height and tilt**

For a proper view of vehicles and thus, to prevent one vehicle from blocking another, the radar should be positioned at a height of 3 meters or more. Depending on the mounting height, the radar should be tilted downwards, but only slightly. Please follow these recommendations.

HEIGHT	TILT
3 m	11 degrees
4 m	15 degrees
5 m	18 degrees
6 m	22 degrees



Additionally, two conditions are important to consider during installation:

1. Install the device on a **straight section of roadway** along the detection distance, meaning between **0 and 25 meters** from the radar. **Avoid installation on curved sections.**
2. Install the device on a section **where vehicles do not stop** along the detection distance. If vehicles come to a complete stop, duplicates may occur.

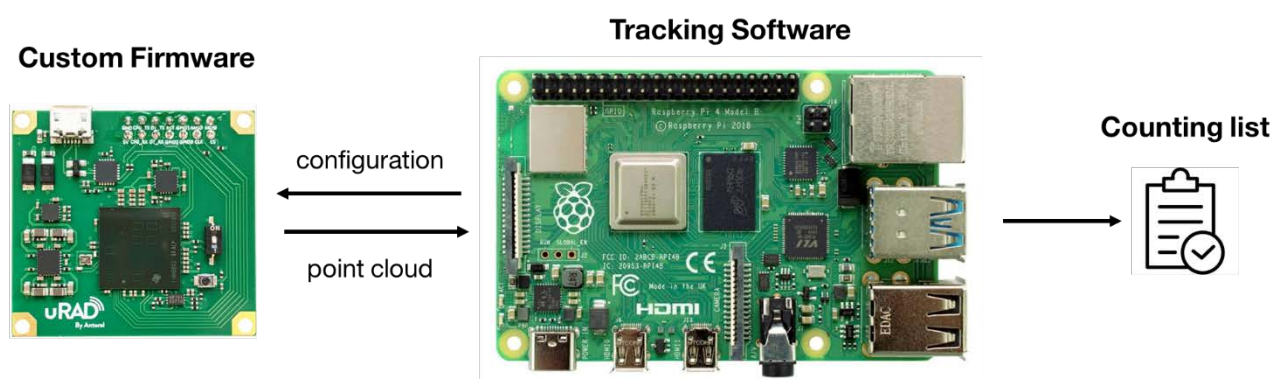
Tracking Software v2.x

The Tracking Software consists of a Python program that has to be run in the master device that controls uRAD Industrial. This software together with the master devices:

- Sends to uRAD Industrial the corresponding configuration parameters.
- Receive from uRAD Industrial the 3D point cloud with X, Y, Z space coordinates, velocity and SNR (Signal to Noise Ratio).
- Process the point cloud to identify vehicles, extract their velocity and classify them.
- Save a counting list with the relevant information.

The target type identifies three types of vehicles according to their length, pedestrians, and bicycles or motorcycles.

Any device where a python program run and that has UART/USB communication can be used as the master device. A typical example of a device that can be easily integrated with uRAD Industrial is a Raspberry Pi.



Tracking software is not included by default with the purchase of uRAD Industrial. Must be purchased separately. In addition, uRAD Industrial carries a custom Firmware that cannot be modified, so that unit will only work for this application. Contact us for more purchasing information.

• Python files

The Tracking Software is comprised of a main Python script and a compiled library.

The main script, named *vehicleCounter_smartcities_v2_x.py*, is open source and it **is the one that has to be run**. The libraries, named *uRAD_Tracking* and *uRAD_Tracking_highway* are compiled and it is where the point cloud data is processed.

The Python version for running the software on Raspberry Pi is 3.9 or 3.11 and on Windows it can be used the version 3.9 or 3.10.

We provide a single script, that can be used with UART communication, assuming, for example, that the radar is used with the PCB adapter + Raspberry Pi, or with USB communication, for instance with a PC. Some configuration parameters have to be set depending on the use case.

- **Mode selection**

Two modes are distinguished: one for high-speed situations where vehicles are expected to travel at speeds exceeding 110 km/h, such as highways, and another for situations where vehicles move at lower speeds.

```
### MODE SELECTION ###
HIGHWAY_SCENARIO = True      # If true the radar operates in high-speed scenario.
```

Set to True for high speed or False for low speed.

Since radar always involves a trade-off between maximum speed and range resolution, the False mode selects an RF configuration better suited for low-speed scenarios with improved resolution, enhancing the precision in vehicle classification.

- **Communication interface**

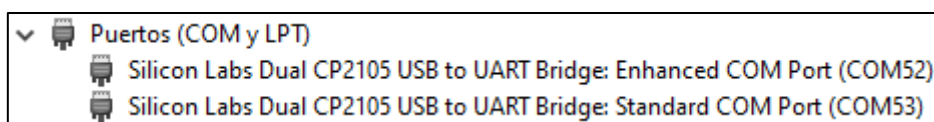
Select True or False if the radar is connected by USB or by UART, respectively.

```
#### COMMUNICATION INTERFACE ####
USB_COMMUNICATION = False
```

Some lines below, set the port names of USB or UART communication.

```
if (USB_COMMUNICATION):
    # Specify serial port names (Enhanced and Standard)
    CONFIG_PORT_NAME = 'COM1'
    DATA_PORT_NAME = 'COM2'
    # When connected to PC you have not access to reset pin, unless you configure
    it manually
    reset = False
else:
    import RPi.GPIO as GPIO
    # Name of serial port (/dev/serial0 in Raspberry Pi)
    PORT_NAME = '/dev/serial0'
    reset = True
```

When uRAD Industrial is connected by USB, two COM ports are recognized. **configPort** is the one identifies as **Enhanced** and **dataPort** is the **Standard**.



If connected by UART and therefore, `USB_COMMUNICATION = False`, by default, the `port_name` is the one for Raspberry Pi (`/dev/serial0`). It is necessary to enable the serial port in *Preferences > Raspberry Pi Configuration > Interfaces* and activate *Serial Port: Enable*, since it is possible that is not enabled by default. The *Serial Console* interface must be disabled so that there is no conflict.

Moreover, it is also assumed that uRAD is used together with the uRAD PCB adaptor for Raspberry Pi, that connects the uRAD Reset pin with GPIO number 6 of Raspberry Pi. If the reset pin is not connected or connected to other RPi pin, modify the lines accordingly.

- **Configuration parameters**

Just a few input configuration parameters have to be set in *vehicleCounter_smartcities_v2_x.py*. At the beginning of the script, there are some lines to introduce the configuration variables:

```
#### CONFIGURATION PARAMETERS ####
USB_CONNECTOR_UPWARD = False
pitch_angle = 15
yaw_angle = 0

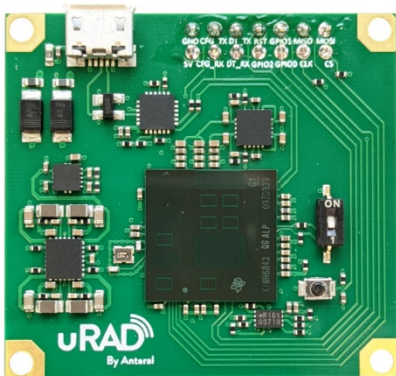
VELOCITY_POSITIVE = True
VELOCITY_NEGATIVE = True

X_MINIMUM_NEGATIVE_VELOCITY = 0
X_MAXIMUM_NEGATIVE_VELOCITY = 4

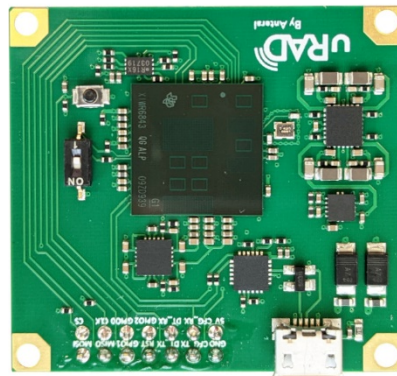
X_MINIMUM_POSITIVE_VELOCITY = 4
X_MAXIMUM_POSITIVE_VELOCITY = 8
```

- **USB_CONNECTOR_UPWARD:** the orientation of the radar is defined by the position of the USB connector. Enter this variable according to the following image.

USB_CONNECTOR_UPWARD = True



USB_CONNECTOR_UPWARD = False

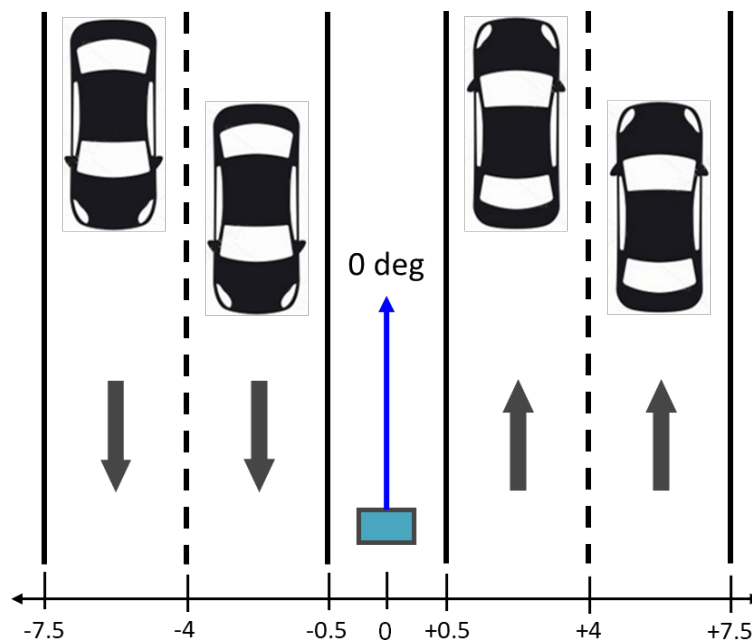


- **pitch_angle:** Defines the tilt angle of the radar with respect to the vertical (in degrees). Enter this parameter according to your installation.

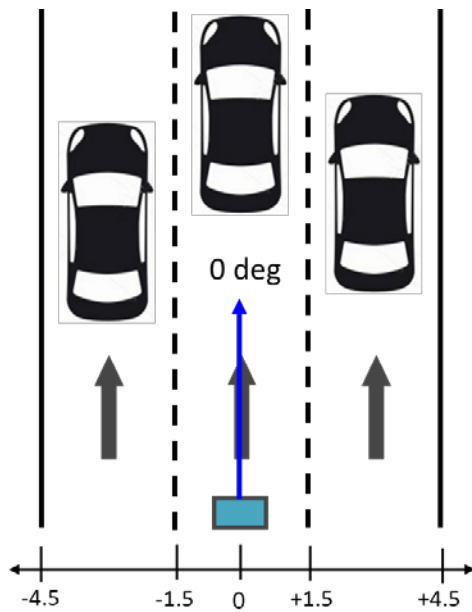
- **yaw_angle:** Defines the aiming angle on the horizontal axis (yaw) with respect to the roadway (in degrees). As mentioned earlier, it is recommended to set the radar at 0 degrees, parallel to the roadway. In case a certain angle is necessary, follow this convention for the angle sign: **to the left of the roadway, positive yaw angle; to the right of the roadway, negative yaw angle.**

- **VELOCITY_POSITIVE:** enter True/False to count/discard vehicles moving away from the radar.
- **VELOCITY_NEGATIVE:** enter True/False to count/discard vehicles approaching the radar.
- **X_MINIMUM_NEGATIVE_VELOCITY:** define the minimum distance in meters in the horizontal direction that you want to consider for counting vehicles with negative velocity (approaching).
- **X_MAXIMUM_NEGATIVE_VELOCITY:** Define the maximum distance in meters in the horizontal direction that you want to consider for counting vehicles with negative velocity (approaching).
- **X_MINIMUM_POSITIVE_VELOCITY:** Define the minimum distance in meters in the horizontal direction that you want to consider for counting vehicles with positive velocity (moving away).
- **X_MAXIMUM_POSITIVE_VELOCITY:** Define the maximum distance in meters in the horizontal direction that you want to consider for counting vehicles with positive velocity (moving away).

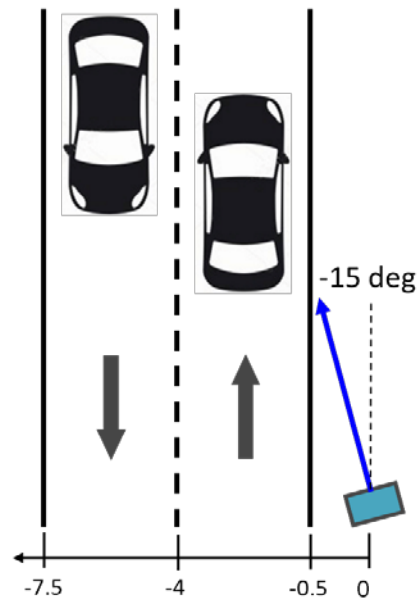
Please refer to the following usage examples.



$X_MINIMUM_NEGATIVE_VELOCITY = -7.5$
 $X_MAXIMUM_NEGATIVE_VELOCITY = -0.5$
 $X_MINIMUM_POSITIVE_VELOCITY = 0.5$
 $X_MAXIMUM_POSITIVE_VELOCITY = +7.5$



X_MINIMUM_NEGATIVE_VELOCITY = not applicable
 X_MAXIMUM_NEGATIVE_VELOCITY = not applicable
 X_MINIMUM_POSITIVE_VELOCITY = -4.5
 X_MAXIMUM_POSITIVE_VELOCITY = +4.5



X_MINIMUM_NEGATIVE_VELOCITY = -7.5
 X_MAXIMUM_NEGATIVE_VELOCITY = -4
 X_MINIMUM_POSITIVE_VELOCITY = -4
 X_MAXIMUM_POSITIVE_VELOCITY = 0

• Output results

User can select to save two types of results:

```
#### OUTPUT RESULTS ####
SAVE_RESULTS = True
SAVE_RAW_DATA =
# files name
FOLDERNAME = 'Results'
OUTPUT_FILENAME = 'Vehicle_results.txt'
POINTCLOUD_FILENAME = 'PointCloud.txt'
```

- **SAVE_RESULTS:** Create a .txt file named **OUTPUT_FILENAME** in the folder named **FOLDERNAME** with the most relevant information. Each line of this text file corresponds to a detected vehicle. The information for each column is as follows:

Timestamp	Velocity	x_distance	Vehicle_type
-----------	----------	------------	--------------

Timestamp: date and time (yyyy/mm/dd HH:MM:SS) or timestamp (UNIX) the vehicle is detected. It is taken from the device system, the Raspberry Pi in this case.

Velocity: velocity in km/h of the vehicle. Positive velocity means vehicle driving away and negative velocity means vehicle approaching.

x_distance: horizontal distance estimation in meters of the vehicle. Useful for lane identification.

Vehicle_type: vehicle type identification. 1 = normal vehicle, 2 = medium vehicle, 3 = large vehicle, 4 = Bicycles/motorcycles, 5 = pedestrian.

- A normal vehicle is any vehicle up to approximately 8 meters in length.
- A medium vehicle is a vehicle between approximately 8 and 15 meters in length.
- A large vehicle is a vehicle over approximately 15 meters in length.
- Bicycles and motorcycles are classified as the same type.
- A pedestrian is any detected target with a speed less than 10 km/h.

- **SAVE_RAW_DATA:** create a .txt file named **POINTCLOUD_FILENAME**. This information is useful for the uRAD team to verify the proper functioning of the radar. This file contains the complete 3D point cloud. Each line starts with the radar frame number and its corresponding timestamp. Subsequently, each line contains (X,Y,Z,velocity,SNR,noise) of all points in that frame.

• Additional Parameters

- **DEBUG_VEHICLES_DEF:** True/False to print or not, in the console, a line for each vehicle with the time, velocity, horizontal distance, and type.

```
New Vehicle: 2024/04/09 10:13:29; X: 5.37 m; Velocity: 32.65 km/h: Type 1
New Vehicle: 2024/04/09 10:13:31; X: 4.92 m; Velocity: 23.71 km/h: Type 1
New Vehicle: 2024/04/09 10:13:34; X: 5.51 m; Velocity: 31.02 km/h: Type 1
New Vehicle: 2024/04/09 10:13:36; X: -6.79 m; Velocity: -26.95 km/h: Type 1
New Vehicle: 2024/04/09 10:13:42; X: -6.66 m; Velocity: -32.57 km/h: Type 1
New Vehicle: 2024/04/09 10:13:50; X: -6.86 m; Velocity: -22.54 km/h: Type 1
...
```

- **USE_FAN:** When the radar is used with a Raspberry Pi, the software is configured to utilize a PWM fan connected to pin 12 of the Raspberry Pi. If an excessive radar temperature is detected, the fan is activated to reduce it.