

## Conteo de Vehículos – Nota de Aplicación – v2.x

### Resumen

Conteo de vehículos es una aplicación que está totalmente resuelta con el modelo uRAD Industrial + Tracking Software. Te permite contar vehículos en múltiples carriles, medir la velocidad y clasificarlos, con gran precisión y mínima configuración. El sistema trabaja en la banda de 60 GHz, una banda de frecuencia para emisión disponible en todo el mundo, lo que facilita la certificación de cualquier producto.

### Caso de Uso

El sistema es muy versátil y se puede usar en muchos escenarios de conteo:

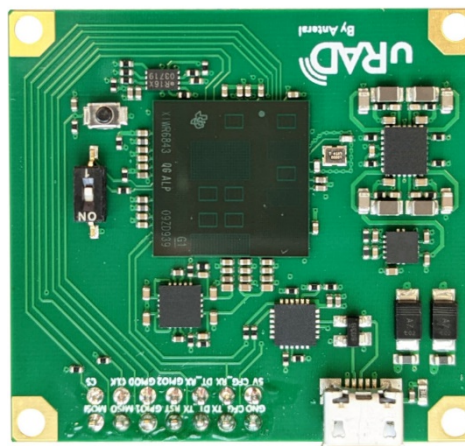
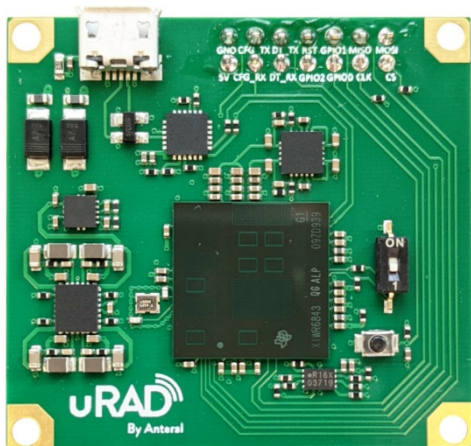
- Carreteras urbanas e interurbanas.
- Medida de velocidad hasta 180 km/h.
- Monitorización de hasta 6 carriles con un solo radar.
- Contar vehículos con velocidad positiva (alejándose) o negativa (acercándose) al mismo tiempo.
- Escenarios de tráfico densos o ligeros.

### Montaje

Hay que tener en cuenta tres aspectos en el montaje del sistema de conteo de vehículos:

- **Orientación del radar**

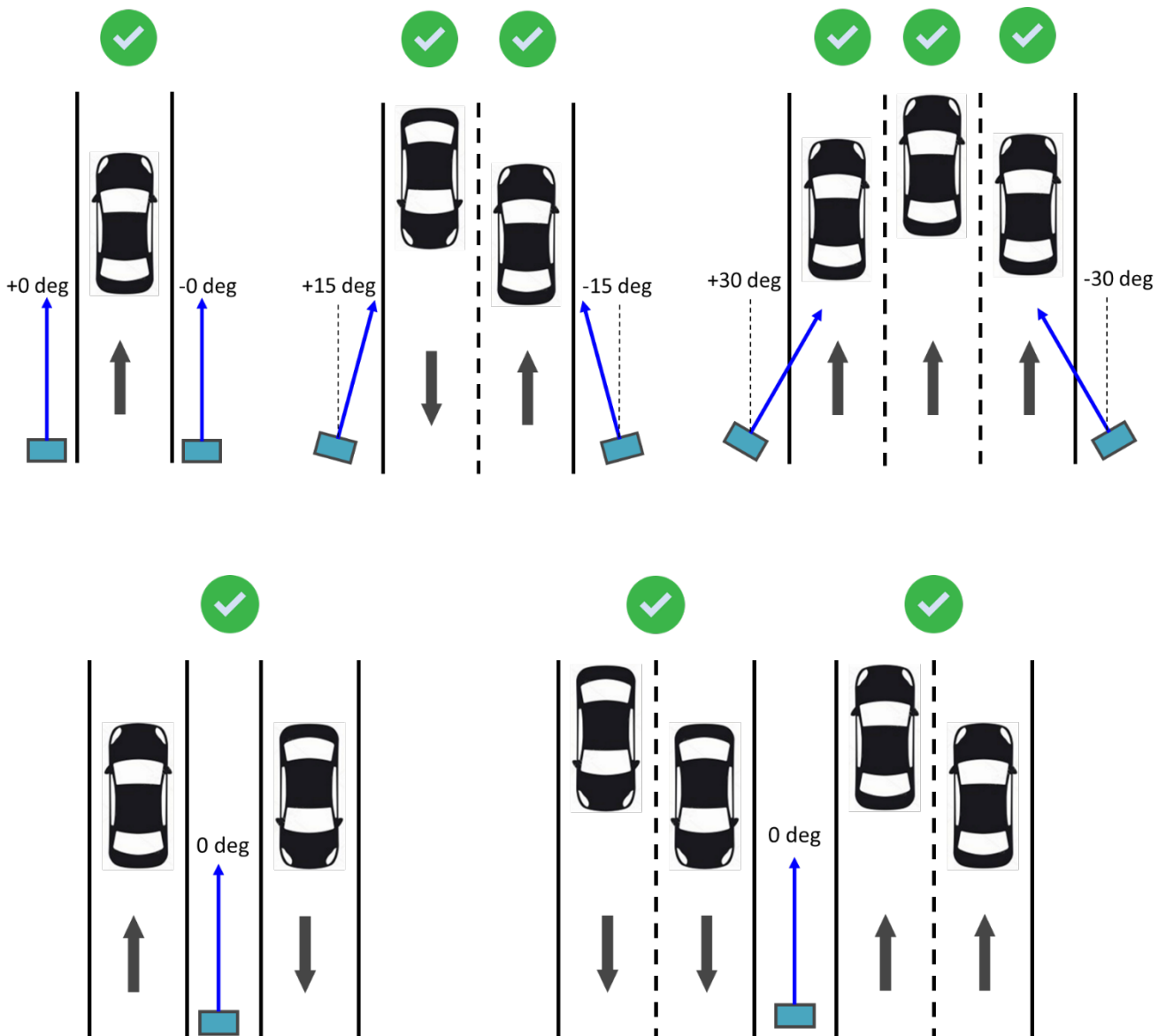
uRAD Industrial se tiene que montar de manera que el conector USB quede en la parte superior o inferior. Cualquiera de las dos opciones es correcta.

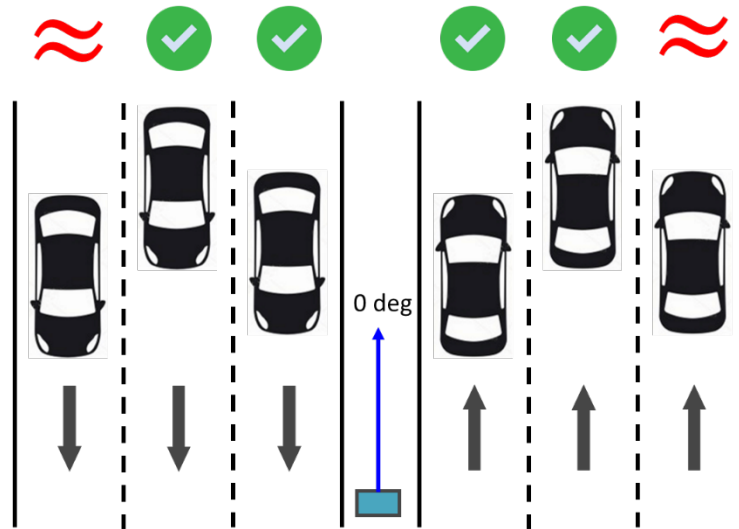


- **Montaje del radar respecto a la carretera**

uRAD Industrial se puede montar a un lado de la carretera, **tanto a la izquierda como a la derecha**, o **encima de ella**. Según el caso de uso, la recomendación general de montaje es la siguiente: para medir **1 carril** a un lado del radar, montar con un ángulo de **0 grados**, para **2 carriles**, con un ángulo de **15 grados** y para **3 carriles**, con un ángulo de **30 grados**.

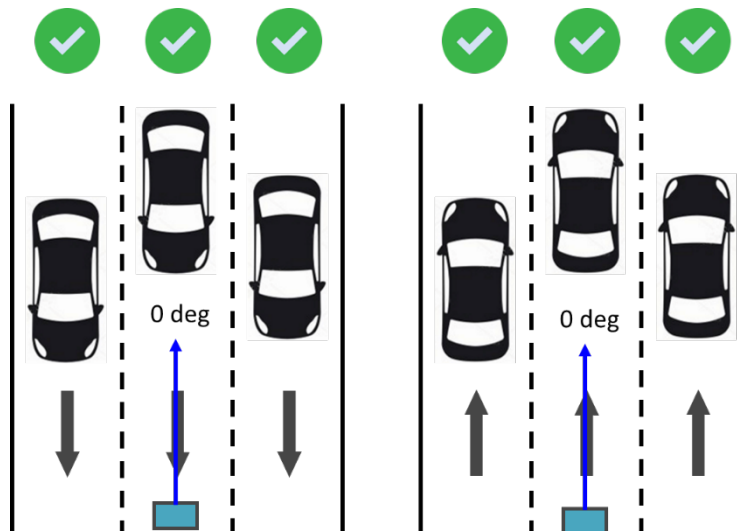
Las siguientes imágenes ilustran los casos de uso principales y como el radar debe ser montado.



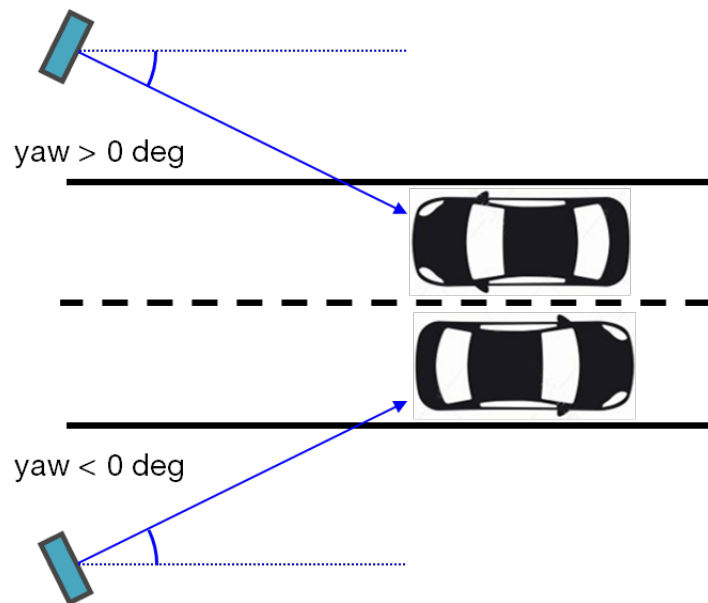


En el caso de uso de 6 carriles, la precisión en el conteo en los carriles de los extremos puede verse comprometida en función del tamaño de los carriles y la densidad de tráfico. Como regla general, el dispositivo ofrece una **buena precisión hasta los 8 metros de distancia lateral**.

Para este caso recomendamos un radar para cada sentido.



En los casos de que sea necesario dar algún ángulo YAW, por ejemplo, en el caso de 2 o 3 carriles o porque algún objeto bloquea el campo de visión, **tenga en cuenta el signo del ángulo** para los parámetros de configuración de acuerdo a la siguiente imagen.



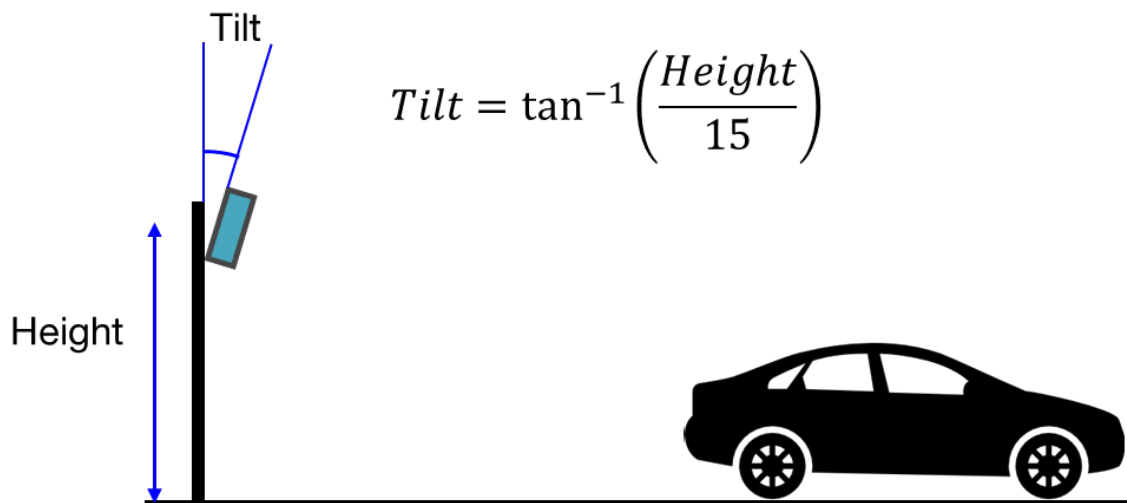
El radar se monta a la **izquierda de la calzada** y por tanto los coches pasan por la derecha del radar (desde un observador colocado detrás del radar) → **ángulo YAW con signo positivo.**

El radar se monta a la **derecha de la calzada** y por tanto los coches pasan por la izquierda del radar (desde un observador colocado detrás del radar) → **ángulo YAW con signo negativo.**

## • Altura del radar e inclinación

Para tener una visión adecuada de los vehículos y, por lo tanto, evitar que un vehículo bloquee a otro, el radar se debe colocar a una altura de 3 metros o más. Dependiendo de la altura de montaje, el radar se debe inclinar hacia abajo, pero solo un poco. Siga estas recomendaciones.

ALTURA (height)	INCLINACIÓN (tilt)
3 m	11 grados
4 m	15 grados
5 m	18 grados
6 m	22 grados



Además, dos condiciones son importantes para tener en cuenta en la instalación:

1. Instalar el dispositivo en un **tramo de carretera recto** a lo largo de la distancia de detección, es decir **entre 0 y 25 metros** desde el radar. **Evita la instalación en tramos con curvas.**
2. Instalar el dispositivo en un **tramo donde los vehículos no se detengan** a lo largo de la distancia de detección. Si los vehículos se detienen completamente, se pueden producir duplicados.

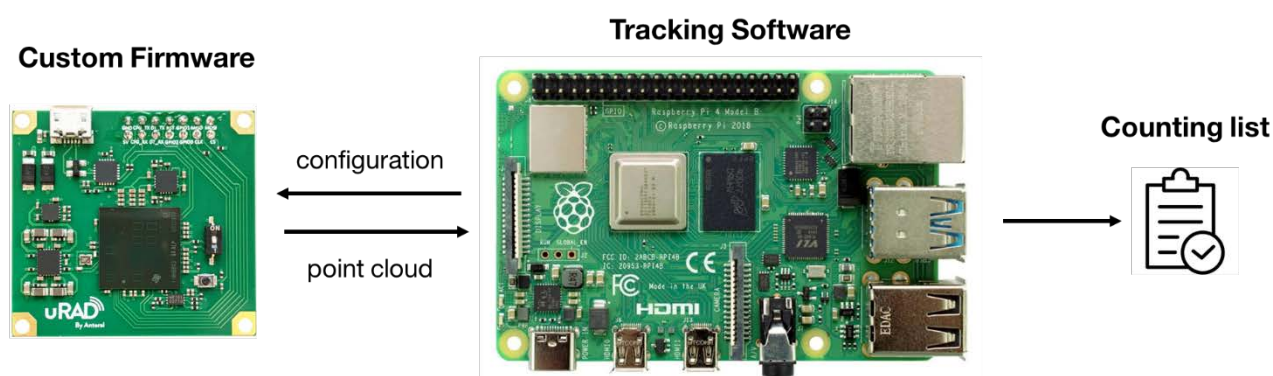
## Tracking Software v2.x

El Tracking Software consiste en un programa de Python que se debe ejecutar en el dispositivo maestro que controla uRAD Industrial. Este software, junto con el dispositivo maestro:

- Envía a uRAD Industrial los correspondientes parámetros de configuración.
- Recibe de uRAD Industrial la nube de puntos 3D con las coordenadas espaciales X, Y, Z, velocidad y SNR (Signal to Noise Ratio).
- Procesa la nube de puntos para identificar vehículos, extrae su velocidad y los clasifica.
- Guarda la lista de conteo con la información relevante.

El tipo de objetivo identifica tres tipos de vehículos de acuerdo a su longitud, peatones y bicicletas o motocicletas.

Cualquier dispositivo que pueda correr Python y tenga comunicación UART/USB puede usarse como dispositivo maestro. Un ejemplo típico de dispositivo que se puede integrar fácilmente con uRAD Industrial es una Raspberry Pi.



**Tracking software no está incluido por defecto con la compra de uRAD Industrial. Debe comprarse por separado. Además, uRAD Industrial lleva un Firmware customizado que no puede modificarse, por lo que esa unidad solo funcionará para esta aplicación. Contáctenos para más información de compra.**

### • Archivos de Python

El Tracking Software está compuesto por un programa principal de Python y dos librerías compiladas.

El programa principal, nombrado *vehicleCounter\_smartcities\_v2\_x.py*, es de código abierto y **es el que se debe lanzar**. Las librerías, nombradas *uRAD\_Tracking* y *uRAD\_Tracking\_highway*, están compiladas y se encargan de procesar la nube de puntos.

La versión de Python para ejecutar el software en Raspberry Pi es la 3.9 o 3.11 y en Windows se puede utilizar la versión 3.9 o 3.10.

Proporcionamos un solo script que puede usarse con comunicación UART, asumiendo, por ejemplo, que el radar se usa con el adaptador PCB + Raspberry Pi, o con comunicación USB, por ejemplo, con un PC. Algunos parámetros de configuración se deben fijar según el caso de uso.

- **Selección de modo**

Se distinguen dos modos, para situaciones de alta velocidad donde se espera que los vehículos puedan circular a velocidades superiores a 110 km/h como autopistas, y para situaciones donde los vehículos circulan a más baja velocidad.

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

Configura True para alta velocidad o False para baja velocidad.

Como en radar siempre existe un compromiso entre velocidad máxima y resolución en rango, el modo False selecciona una configuración RF más adaptada para baja velocidad y mejor resolución, donde se aumenta la precisión en la clasificación de vehículos.

- **Interfaz de comunicación**

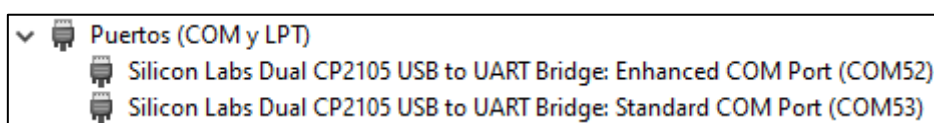
Selecciona True o False si el radar está conectado por USB o por UART, respectivamente.

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

Algunas líneas abajo, escribe los nombres de los puertos para comunicación USB o UART.

```
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
```

Cuando uRAD Industrial se conecta por USB, se reconocen dos puertos COM. **configPort** es el que se identifica como **Enhanced** y **dataPort** es el **Standard**.



Si conectas por UART y entonces, `USB_COMMUNICATION = False`, por defecto, `PORT_NAME` es el de Raspberry Pi (`/dev/serial0`). Es necesario habilitar el puerto serie en *Preferences*

> *Raspberry Pi Configuration* > *Interfaces* y activar *Serial Port: Enable*, ya que es posible que no esté activado por defecto. La interfaz *Serial Console* debe estar deshabilitada para que no haya conflicto.

Además, se asume que uRAD se usa junto con el Adaptador PCB para Raspberry Pi, que conecta el pin Reset de uRAD con el GPIO número 6 de la Raspberry Pi. Si el pin de Reset no se conecta o se conecta a otro pin de la RPi, modifica la línea en consecuencia.

## • Parámetros de configuración

Solo hay que fijar unos pocos parámetros de entrada en *vehicleCounter\_smartcities\_v2\_x.py*. Al comienzo del programa, hay unas líneas para introducir las variables de configuración:

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

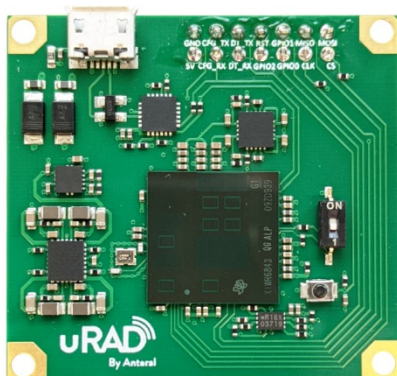
VELOCITY_POSITIVE = True
VELOCITY_NEGATIVE = True

X_MINIMUM_NEGATIVE_VELOCITY = 0
X_MAXIMUM_NEGATIVE_VELOCITY = 4

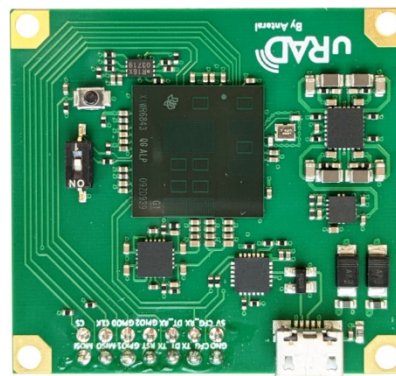
X_MINIMUM_POSITIVE_VELOCITY = 4
X_MAXIMUM_POSITIVE_VELOCITY = 8
```

- **USB\_CONNECTOR\_UPWARD:** define la orientación del radar mediante la posición del conector USB. Introduce esta variable de acuerdo a la siguiente imagen.

**USB\_CONNECTOR\_UPWARD = True**



**USB\_CONNECTOR\_UPWARD = False**

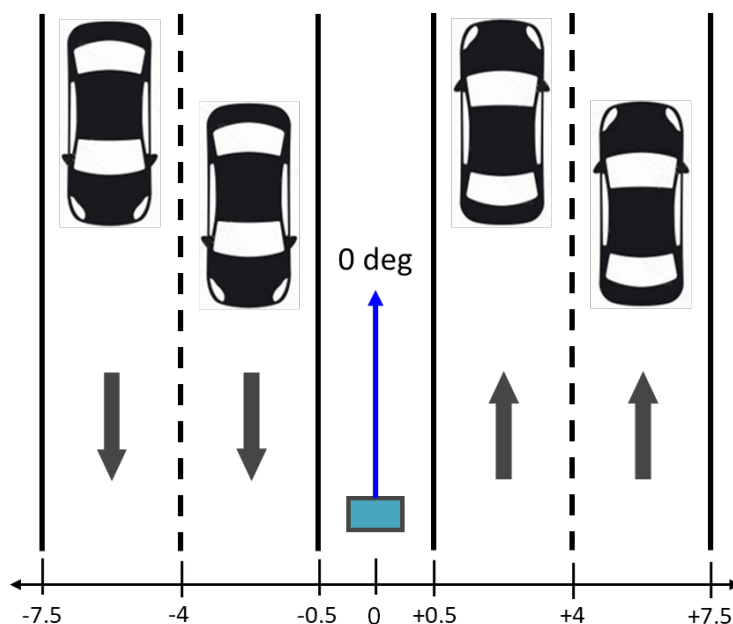


- **pitch\_angle:** define el ángulo de inclinación del radar con la vertical (en grados). Introduce este parámetro de acuerdo a tu montaje.
- **yaw\_angle:** define el ángulo de apuntamiento en el eje horizontal (guiñada) con respecto a la carretera (en grados). Como se ha comentado anteriormente, se recomienda poner el radar con 0 grados, es decir, paralelo a la carretera. En el caso de que fuese necesario

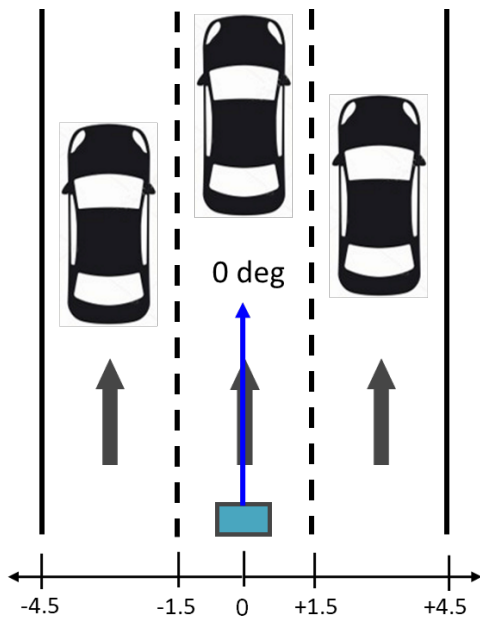
dar un cierto ángulo siga esta convención del signo del ángulo: **a la izquierda de la carretera ángulo yaw positivo, a la derecha de la carretera ángulo yaw negativo.**

- **VELOCITY\_POSITIVE:** introduce True/False para contar/descartar vehículos alejándose del radar.
- **VELOCITY\_NEGATIVE:** introduce True/False para contar/descartar vehículos acercándose al radar.
- **X\_MINIMUM\_NEGATIVE\_VELOCITY:** define la **mínima distancia** en metros en la dirección horizontal que quiere considerar para contar vehículos con **velocidad negativa (acercándose)**.
- **X\_MAXIMUM\_NEGATIVE\_VELOCITY:** define la **máxima distancia** en metros en la dirección horizontal que quiere considerar para contar vehículos con **velocidad negativa (acercándose)**.
- **X\_MINIMUM\_POSITIVE\_VELOCITY:** define la **mínima distancia** en metros en la dirección horizontal que quiere considerar para contar vehículos con **velocidad positiva (alejándose)**.
- **X\_MAXIMUM\_POSITIVE\_VELOCITY:** define la **máxima distancia** en metros en la dirección horizontal que quiere considerar para contar vehículos con **velocidad positiva (alejándose)**.

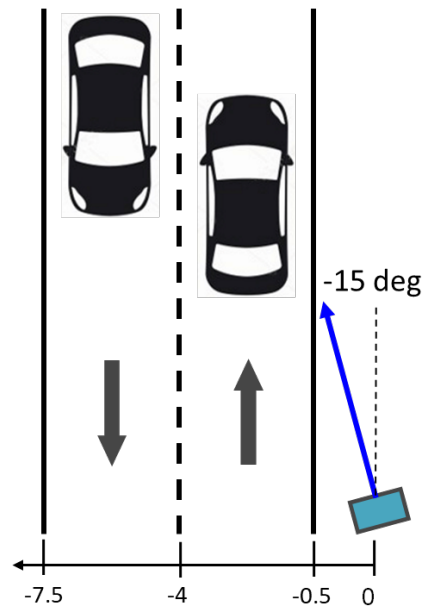
Observa los siguientes ejemplos de casos de uso.



$X\_MINIMUM\_NEGATIVE\_VELOCITY = -7.5$   
 $X\_MAXIMUM\_NEGATIVE\_VELOCITY = -0.5$   
 $X\_MINIMUM\_POSITIVE\_VELOCITY = 0.5$   
 $X\_MAXIMUM\_POSITIVE\_VELOCITY = +7.5$



X\_MINIMUM\_NEGATIVE\_VELOCITY = no aplica  
 X\_MAXIMUM\_NEGATIVE\_VELOCITY = no aplica  
 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

## • Resultados de salida

El usuario puede seleccionar dos tipos de resultados:

```

#### OUTPUT RESULTS ####
SAVE_RESULTS = True
SAVE_RAW_DATA = False
# files name
FOLDERNAME = 'Results'
OUTPUT_FILENAME = 'Vehicle_results.txt'
POINTCLOUD_FILENAME = 'PointCloud.txt' # Name of radar pointCloud file
  
```

- **SAVE\_RESULTS:** crea un .txt nombrado **OUTPUT\_FILENAME**, en la carpeta de nombre **FOLDERNAME** con la información más relevante. Cada línea de este archivo de texto corresponde con un vehículo detectado. La información de cada columna es la siguiente:

**Marca\_temporal**      **Velocidad**      **Distancia\_x**      **Tipo\_vehiculo**

**Marca\_temporal:** fecha y hora (yyyy/mm/dd HH:MM:SS) en la que el vehículo es detectado. Se toma del sistema del dispositivo, la Raspberry Pi en este caso.

**Velocidad:** velocidad en km/h del vehículo. Velocidad positiva significa vehículo alejándose y negativa vehículo acercándose.

**Distancia\_x:** estimación de la distancia horizontal en metros del vehículo. Útil para identificación de carril.

**Tipo\_vehículo:** identificación del tipo de vehículo. 1 = vehículo normal, 2 = vehículo medio, 3 = vehículo largo, 4 = bicicleta/motocicleta, 5 = peatón.

- Un vehículo normal es todo vehículo hasta 8 metros de longitud aproximadamente.
  - Un vehículo medio es un vehículo entre 8 y 15 metros de longitud aproximadamente.
  - Un vehículo largo es un vehículo de más de 15 metros de longitud aproximadamente.
  - Bicicletas y motocicletas se clasifican como el mismo tipo.
  - Un peatón es todo objetivo detectado con una velocidad inferior a 10 km/h.
- **SAVE\_RAW\_DATA:** crea un .txt nombrado **POINTCLOUD\_FILENAME**. Esta información es útil para el equipo de uRAD para comprobar el correcto funcionamiento del radar.

Este archivo contiene la nube de puntos 3D completa. Cada línea comienza con el número de trama del radar y su marca temporal correspondiente. Posteriormente cada línea contiene (X,Y,Z,velocidad,SNR,ruido) de todos los puntos de esa trama.

### • Parámetros adicionales

- **DEBUG\_VEHICLES\_DEF:** True/False para imprimir o no, en la consola, una línea por cada vehículo con el tiempo, velocidad, distancia horizontal y tipo.

```
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:** el software, cuando el radar se usa junto a la Raspberry Pi, está preparado para usar un ventilador de tipo PWM conectado al pin 12 de la Raspberry Pi. Si se detecta una temperatura del radar excesiva, el ventilador se enciende para disminuirla.