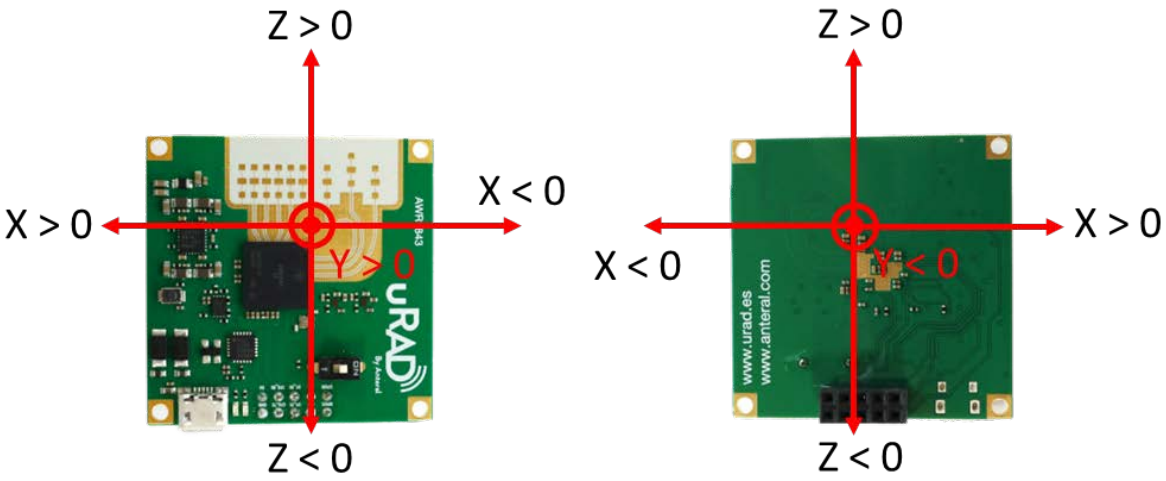
Short Range Radar – Application Notes

Overview

Short Range Radar firmware and software demonstrates the ability of uRAD Automotive HPA to work as a short range radar to detect obstacles up to 10 meters and a field of view of 140 degrees. The obstacles are positioned in the 2D space (range and azimuth) where the height information is omitted.

Radar position

The radar axis is defined according to the following image.



Notice that the **point cloud information does not contain Z-axis information** because the obstacles are positioned in the XY plane.

Therefore, radar orientation should be:

- X axis parallel to ground (±70 degrees azimuth field of view)
- Y axis pointing to front

Firmware and chirp configuration

The firmware used is the Out of Box Demo, the regular firmware to obtain the point cloud based on a desired chirp configuration. The chirp configuration has been chosen to maximize the range and angle accuracy at short distances.

Parameter	Value	Details
Max. Range	10 m	Maximum distance that the radar can detect an object.
Range Resolution	4 cm	Capability to distinguish between two or more targets on the same bearing but at different ranges.
Max. Velocity	4.7 m/s	Maximum velocity obtained.
Velocity Resolution	0.3 m/s	Capability to distinguish between two or more targets at the same range but moving with different velocities.
Angle Accuracy	1 degree	Precision to position in angle the detected object.
Frame rate	30 fps	Number of detections per second.

The radar sends the point cloud. The point cloud contains the information of (X,Y,Z,velocity, SNR,noise) of detected points. Each detected object generates one or more points in the point cloud. The number of points each object generate depends on many factors (position, velocity, shape, materials, etc.).

- Although the maximum velocity is 4.7 m/s = 17 km/h, the radar is able to detect the objects which are moving at much more velocity due to an extended velocity mode.
- The value of Z is always 0 because the chirp configuration is selected to maximize the accuracy in the 2D plane.

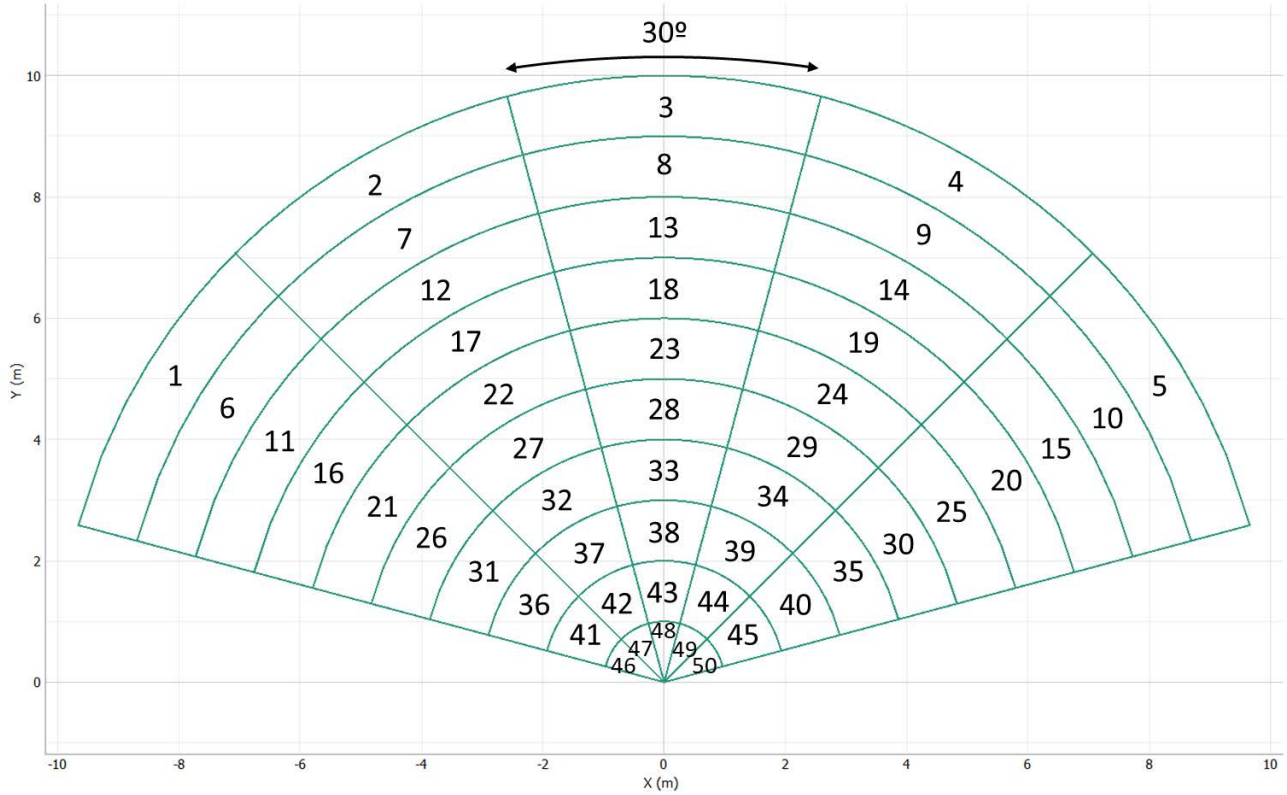
Detection

The key point of this application is that the front area of the radar is divided in **50 detection zones**. These detection zones have a size of **1 meter by 30 degrees**.

If one or more points are detected inside each zone during more than 3 consecutive frames, the zone is active and we can conclude that there is and object occupying that zone.

This condition of more than 3 consecutive frames is implemented to reduce false positives caused by clutter.

In the generated results, the zones are saved as a matrix of 10 rows and 5 columns. Each position of the matrix corresponds with a zone, according the following picture.



If there is and object in the zone, the corresponding position in the matrix is set to 1, if there is NOT object, the value of the position is 0.

Matrix of detected zones

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30
31	32	33	34	35
36	37	38	39	40
41	42	43	44	45
46	47	48	49	50

Software

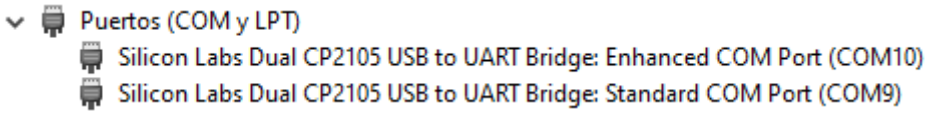
Two python scripts have been programmed to work with the radar. These scripts send the chirp configuration to radar, receive the point cloud and create the matrix of detected zones. Both, the point cloud and the matrix can be saved in .txt files.

- **short_range_radar_USB.py**

This script is for using uRAD Industrial connected by USB. In the script, the COM ports where the radar is connected have to be set.

```
configPort_name = 'COM2'
dataPort_name = 'COM1'
```

COM port are easily identified in with the *Device Manger*:



- Enhanced port is the configuration port
- Standard port is the output data port

- **short_range_radar_single_UART.py**

This script is for using uRAD Automotive HPA by the pin connector using a single UART channel. For example, this script is useful when using uRAD together with the PCB adapter for Raspberry Pi. The port name has to be introduced. For instance, in the Raspberry Pi is:

```
Port_name = '/dev/serial0'
```

In this script is also available the possibility of doing a reset using the Reset pin of the pin connector.

```
reset = True
```

The pin name/number of the master device where the uRAD reset pin is connected has to be set. Using the PCB adapter this pin is GPIO number 5.

```
        PinReset = OutputDevice(5)
```

Two parameters can be configured to limit the range and azimuth angel. In this way, the point cloud is limited and the user can focus on searching objects in the straight direction or shorter range.

```
### CONFIGURATION PARAMETERS ###
max_azimuth = 90    # View angle of the radar. Possible values are 30, 90 or 150
max_range = 8 # Detection max range of the radar. Possible values are 4, 6, 8 or 10
```

There are only allowed some values for range and azimuth:

```
possible_ranges = [4,6,8,10]
possible_azimuth = [30,90,150]
if max_range not in possible_ranges or max_azimuth not in possible_azimuth:
    sys.exit("Range or value not set corretly")
```

Results

The point cloud and zone information are saved in two different .txt. whether the corresponding variable is set to True.

```
savePointCloud = True
saveDetectionZones = True
```

And their names.

```
pointCloud_fileName = './output_files/PointCloud.txt'
detectedZones_fileName = './output_files/DetectedZones.txt'
```

- **PointCloud.txt**

Each line of this file contains the information of the point cloud in each frame. It is saved consecutively in each column:

[X (m)] [Y (m)] [Z (m)] [Velocity (m/s)] [SNR] [Noise]

of every detected point of the point cloud. At the end, the timestamp is included, as the number of seconds since January 1, 1970, 00:00:00 (UTC).

- **DetectedZones.txt**

Each line of this file contains the information of the full matrix in each frame. It saved consecutively in each column:

[1] [2] [3] … [50] →each position saves 0 (NO obstacle), 1 (YES obstacle)

At the end, the timestamp is included, as the number of seconds since January 1, 1970, 00:00:00 (UTC).

Visualizer

A visualizer has been also programmed to ease the visualization of the results in real time when using the radar by USB.

The visualize can be found in the folder *GUI*. Run the python script **short_range_radar_GUI.py** to use it.

With this visualizer:

1. Limit the results in range and azimuth angle according to the configuration parameters.
2. Save the point cloud .txt file (zone matrix cannot be saved)
3. Re-play previously saved results.

In this visualizer, the point cloud is drawn as black dots and the zones are colored when they are active and therefore, there is an obstacle in it.