

# mmWave sensors in robotics: technical deep dive

# Detailed agenda

- mmWave Sensing in Robotics – how do robots “see” using mmWave?
  - Overview and market differentiation
  - mmWave Demo Visualizer
  - ROS (Robot OS) Point Cloud Visualizer lab on TI Resource Explorer
- Autonomous robot demonstration using ROS + TI mmWave sensor (IWR1443)
- Technical Deep Dive
  - Tuning the mmWave sensor configuration for a specific application
  - How the “Autonomous Robotics with ROS for mmWave” demo works
  - Tuning Robot OS parameters in the demo

## Technical deep dive

- **Tuning the mmWave sensor configuration for a specific application**
- How the “Autonomous Robotics with ROS for mmWave” demo works
- Tuning Robot OS parameters in the demo

# Tuning the mmWave sensor configuration for a specific application

- Goal is to create a mmWave sensor chirp configuration that satisfies the sensing needs of the application

Development Stage	Method
Discover/Evaluate ↓	<b>mmWave Demo Visualizer</b> can be used to create and test a chirp configuration that will work with the mmWave SDK out-of-box demo which is used in the “Autonomous Robotics with ROS for mmWave” lab
Evaluate/Design	<b>mmWave Sensing Estimator</b> can be used to create more advanced/customized chirp configurations that may not work with the out-of-box demo and instead require a custom processing chain

# Creating a chirp configuration with mmWave Demo Visualizer

Steps to create a chirp configuration:

- Select the desired parameters on the Configure tab in the mmWave Demo Visualizer
- Verify that the config works by sending the config to the mmWave device and review the live plots on the Plots tab
- Press the “Save Config to PC” button to save the chirp config to a file
- If you are on a Windows computer, copy the chirp config file to the Linux PC connected to the mmWave EVM

The screenshot shows the mmWave Demo Visualizer software interface. The top bar is red with the text "mmWave Demo Visualizer" and "Options Help". Below this is a dark blue bar with the text "Configure". The main area is divided into several sections:

- Setup Details:** Platform (xWR14xx), SDK version (1.2), Antenna Config (4Rx,3Tx(15 deg + Elevation)).
- RCS:** Desired Radar Cross Section (sq. m) (0.5), Maximum Range for desired RCS (m) (53.506), RCS at Max Unambiguous Range (sq. m) (0.000402).
- Desirable Configuration:** Best Range Resolution, Frequency Band (GHz) (77-81).
- Scene Selection:** Frame Rate (fps) (10), Range Resolution (m) (0.044), Maximum Unambiguous Range (m) (9.01), Maximum Radial Velocity (m/s) (1), Radial Velocity Resolution (m/s) (0.13).
- Object Detection:** Group Peaks from Same Object (checked), Range Direction (checked), Doppler Direction (checked), Additional Algorithm Processing (Remove Static Clutter), Range Detection Threshold (0-100dB) (15).
- Range/Angle Bias Compensation config:** compRangeBiasAndRxChanPhase 0.0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
- Plot Selection:** Scatter Plot (checked), Range Profile (checked), Noise Profile (unchecked), Range Azimuth Heat Map (unchecked), Range Doppler Heat Map (unchecked), Statistics (checked).

At the bottom, there are three buttons: "SEND CONFIG TO MMWAVE DEVICE", "SAVE CONFIG TO PC" (highlighted with a red box), and "RESET SELECTION".

## Using the chirp configuration from the mmWave Demo Visualizer

- Chirp configurations generated from the mmWave Demo Visualizer can be used directly by the Robot OS mmWave labs
- It is possible to reconfigure the mmWave sensor after it is already running as long as the previous config used the same number of TX and RX antennas (command is given in ROS Point Cloud Visualizer user guide)
- It is also possible to replace the default config so that the new config is loaded when the lab is started (required if the number of TX/RX antennas differs)
  - Place the new chirp config file in the “~/catkin\_ws/src/ti\_mmwave\_ropkg/cfg” directory
  - Rename the original default config file to a different name
  - Rename the new config file to match the original default config file

# Creating a chirp configuration with mmWave Sensing Estimator

- mmWave Sensing Estimator can be used to create more advanced / customized chirp configurations
- Inputs
  - Device type and number of antennas
  - Board-specific antenna gains
  - Regulatory Restrictions
  - Scene Parameters
  - Additional Parameters
- Outputs
  - Chirp Configuration Parameters
  - Information Only Parameters
  - Detectable Object Range (also for information only)
  - Errors (if config is not valid)

**mmWave Sensing Estimator** Help

**Assumptions and Inputs** Short Range Default

**Device Specific Parameters**

mmWave Sensor: IWR1443

# 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): 77 - 81

Maximum Bandwidth (MHz): 4000

Transmit Power (dBm): 12

**Scene Parameters**

Ambient Temperature (deg Celcius): 20

Maximum Detectable Range (m): 10

Range Resolution (cm): 5

Maximum Velocity (km/h): 26

Velocity Resolution (km/h): 2

Measurement Rate (Hz): 10

Typical Detectable Object (m<sup>2</sup>): Adult

**Additional Parameters**

Detection Loss (dB): 1

System Loss (dB): 1

Implementation Margin (dB): 2

Detection SNR (dB): 12

**Outputs and Chirp Design**

Detectable Object Range

Max Range for Typical Detectable Object (m): 52.04

Min RCS Detectable at Max Range (m<sup>2</sup>): 0.00

**Chirp Configuration Parameters**

Frequency Start (GHz)	77	# of Chirp Loops	27
Frequency Slope (MHz/us)	58.13	Frame Periodicity (ms)	100
Frequency Slope Constant	1204	Idle Time (us)	7
Sampling Rate (ksp/s)	4306	ADC Valid Start Time (us)	6.30
# of Samples per Chirp	223	Ramp End Time	59.09

**Information Only Parameters**

Bandwidth (MHz)	3434.84	# of Range FFT Bins	256
Beat Frequency (MHz)	3.88	# of Doppler FFT Bins	32
Chirp Cycle Time (us)	66.09	Range Interbin Resolution (cm)	4.36
Chirp Repetition Period (us)	132	Velocity Interbin Resolution (m/s)	0.47
Active Frame Time (ms)	3.56	Radar Cube Size (KB)	216

No errors are found.

SAVE CONFIG LOAD CONFIG RESET CONFIG

## Using the chirp configuration from the mmWave Sensing Estimator

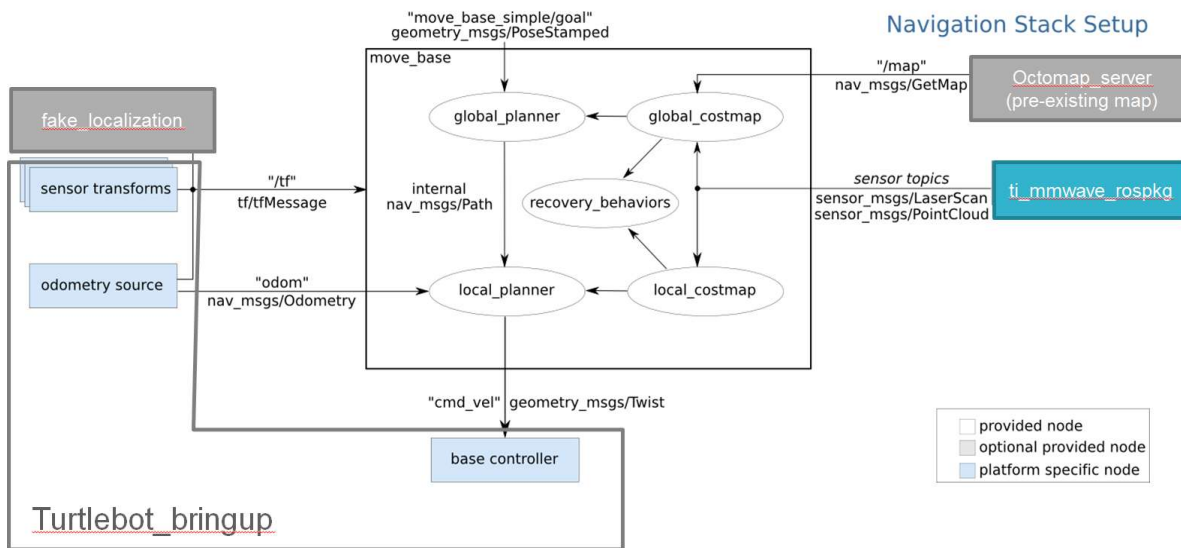
- Chirp configurations generated from the mmWave Sensing Estimator may require a custom processing chain
- In that case, the mmWave SDK out-of-box demo code would need to be modified as follows:
  - If the command-line interface (CLI) rejects the new chirp config parameters, it would need to be modified
  - The mmWave SDK processing chain would need to be modified to support the dataflow, timing, and signal processing requirements for the new chirp configuration
- In order to work with the Robot OS mmWave labs, the modified mmWave SDK demo must still output the same detected object data format over the UART



## Technical deep dive

- Tuning the mmWave sensor configuration for a specific application
- **How the “Autonomous Robotics with ROS for mmWave” demo works**
- Tuning Robot OS parameters in the demo

# How the “Autonomous Robotics with ROS for mmWave” demo works



- ROS move\_base navigation package
  - Global costmap/planner to plot path
  - Local costmap/planner to account for robot movement capabilities
  - Added recovery behavior to clear obstacle map and re-scan when no path found
- ROS fake\_localization package
  - Allows user to set initial location and desired destination on map
- Sensors
  - Odometry to track location
  - TI mmWave for obstacle detection
- Map server
  - Made static so it can be used to define constrained area for robot to stay within

## Technical deep dive

- Tuning the mmWave sensor configuration for a specific application
- How the “Autonomous Robotics with ROS for mmWave” demo works
- **Tuning Robot OS parameters in the demo**

# Tuning Robot OS parameters in the demo

- ROS navigation stack is described at: <http://wiki.ros.org/navigation>
- In the demo, Robot OS navigation stack parameter files are located at:  
~/catkin\_ws/src/turtlebot/turtlebot\_apps/turtlebot\_navigation/param/
- Parameter files:
  - costmap\_common\_params.yaml
  - global\_costmap\_params.yaml
  - global\_planner\_params.yaml, navfn\_global\_planner\_params.yaml
  - local\_costmap\_params.yaml
  - dwa\_local\_planner\_params.yaml
  - move\_base\_params.yaml
  - radar\_costmap\_params.yaml (used to override settings in other param files)

# Important tuning parameters for local planner

- dwa\_local\_planner\_params.yaml
  - Parameters described at: [http://wiki.ros.org/dwa\\_local\\_planner?distro=kinetic](http://wiki.ros.org/dwa_local_planner?distro=kinetic)
  - max\_vel\_x = maximum forward velocity (m/s)
  - acc\_lim\_x = maximum forward acceleration (m/s<sup>2</sup>)
  - max\_rot\_vel = maximum rotational velocity (rad/s)
  - acc\_lim\_theta = maximum rotational acceleration (rad/s<sup>2</sup>)
  - xy\_goal\_tolerance = tolerance (in meters) when trying to reach goal

# Important tuning parameters for costmaps

- costmap\_common\_params.yaml (radar\_costmap\_params.yaml overrides the global costmap inflation\_layer parameters)
  - Parameters described at: [http://wiki.ros.org/costmap\\_2d?distro=kinetic](http://wiki.ros.org/costmap_2d?distro=kinetic) (in section “8.2 Layer Specifications”)
  - robot\_radius = radius of robot (m)
  - obstacle\_layer
    - z\_voxels = number of cells in z-axis in 3D costmap occupancy “voxel” grid (max = 16)
    - z\_resolution = z resolution of 3D occupancy “voxel” grid (meters/cell), height of the grid is  $z\_resolution * z\_voxels$
    - mark\_threshold = maximum number of marked cells allowed in a column considered to be “free” (i.e. if more cells in column are marked, then the X/Y grid location is considered to be occupied)
    - obstacle\_range = maximum distance from the robot at which an obstacle will be inserted into the cost map (m)
  - inflation\_layer
    - inflation\_radius = max distance from an obstacle at which costs are incurred for planning paths
    - cost\_scaling\_factor = exponential rate at which the obstacle cost drops off

## Customer collateral

Content type	Content title	Link to content or more details
Labs on TI CCS Resource Explorer	ROS Point Cloud Visualizer lab and Autonomous Robotics with ROS for mmWave lab	<a href="http://dev.ti.com/tirex/#/?link=Software%2FmmWave%20Sensors%2FIndustrial%20Toolbox">http://dev.ti.com/tirex/#/?link=Software%2FmmWave%20Sensors%2FIndustrial%20Toolbox</a> (under Labs)
Customer training series	mmWave Training Series	<a href="https://training.ti.com/mmwave-training-series">https://training.ti.com/mmwave-training-series</a>
Technical blog content or white paper	mmWave radar sensors in robotics applications	<a href="http://www.ti.com/lit/wp/spry311/spry311.pdf">http://www.ti.com/lit/wp/spry311/spry311.pdf</a>
Selection and design tools and models	mmWave Sensing Estimator	<a href="https://dev.ti.com/mmWaveSensingEstimator">https://dev.ti.com/mmWaveSensingEstimator</a>
	mmWave Demo Visualizer	<a href="https://dev.ti.com/mmWaveDemoVisualizer">https://dev.ti.com/mmWaveDemoVisualizer</a>
Videos	mmWave Demo Visualizer video	<a href="https://training.ti.com/mmwave-sdk-evm-out-box-demo">https://training.ti.com/mmwave-sdk-evm-out-box-demo</a>
	ROS Point Cloud Visualizer lab video	<a href="https://youtu.be/INEGT10Mk9k">https://youtu.be/INEGT10Mk9k</a>
	Autonomous Robotics with ROS for mmWave lab video	<a href="https://training.ti.com/robotics-sense-and-avoid-demonstration-using-ti-mmwave-sensors">https://training.ti.com/robotics-sense-and-avoid-demonstration-using-ti-mmwave-sensors</a>
Product and EVM pages	IWR1443 product page	<a href="http://www.ti.com/product/IWR1443">http://www.ti.com/product/IWR1443</a>
	IWR1443BOOST EVM page	<a href="http://www.ti.com/tool/IWR1443BOOST">http://www.ti.com/tool/IWR1443BOOST</a>
	IWR1642 product page	<a href="http://www.ti.com/product/IWR1642">http://www.ti.com/product/IWR1642</a>
	IWR1642BOOST EVM page	<a href="http://www.ti.com/tool/IWR1642BOOST">http://www.ti.com/tool/IWR1642BOOST</a>