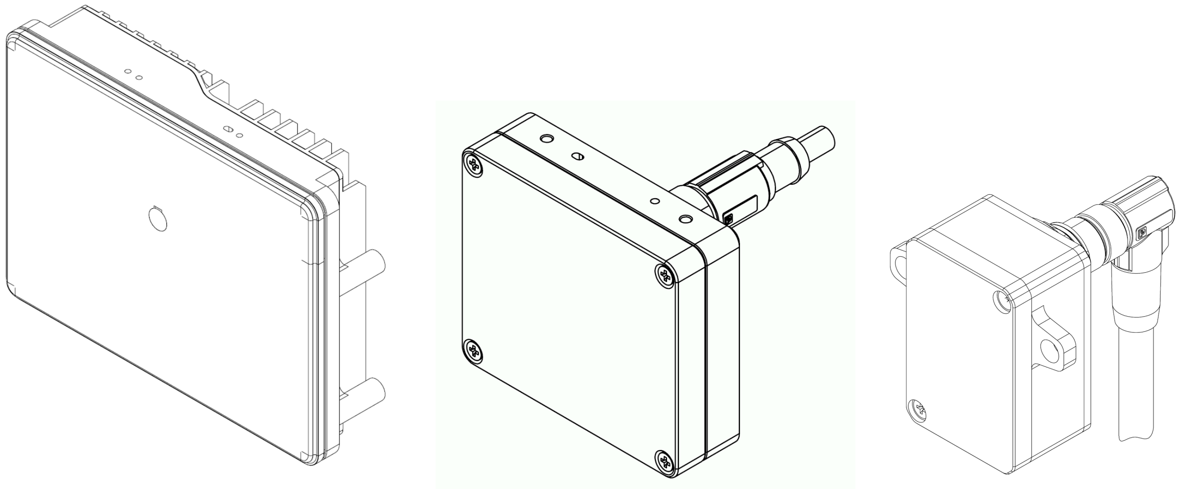


Software Integration Manual

Release v2.4

January 2024

Zadar Radar Platforms

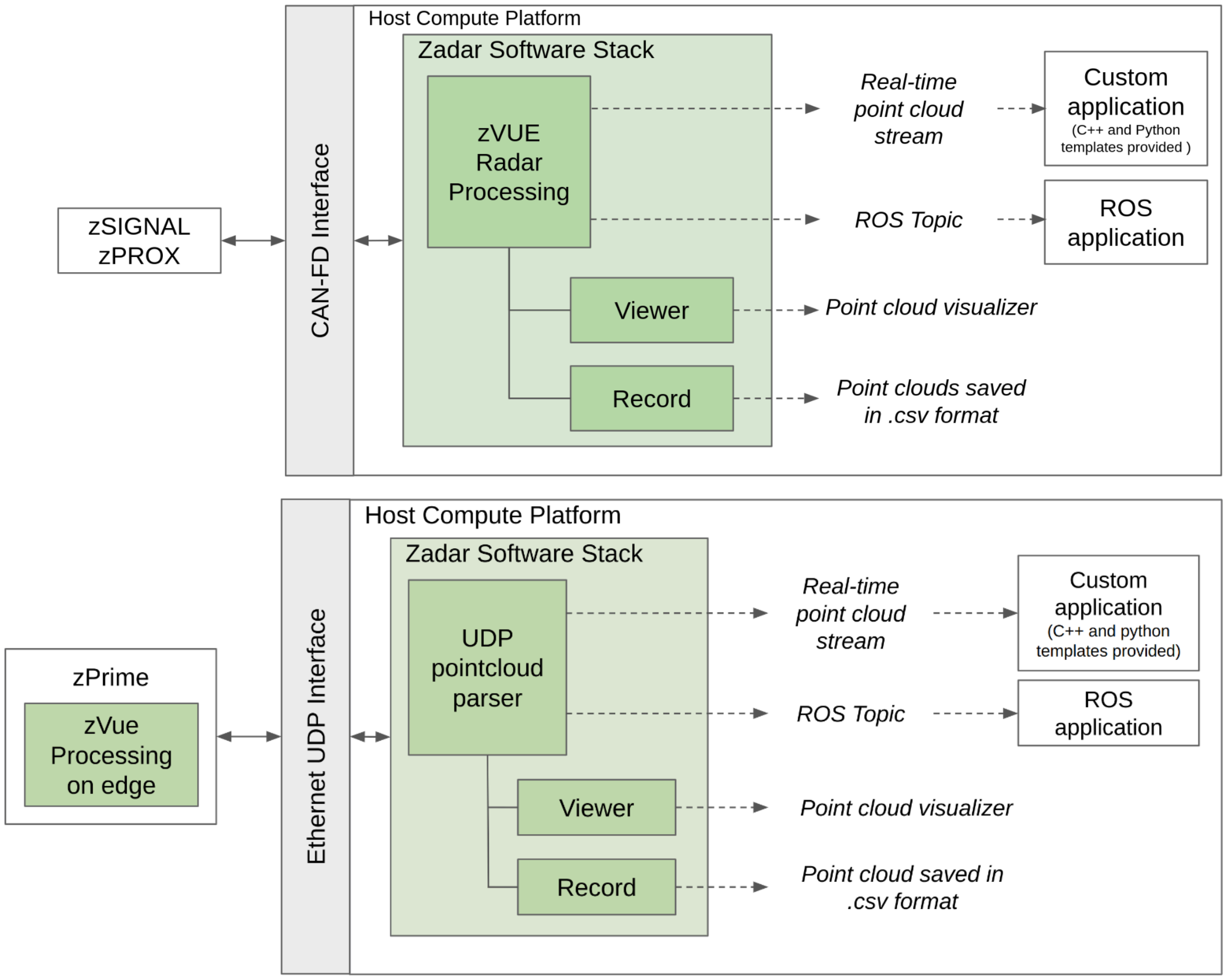
Table of Contents

1 Introduction

This guide discusses the installation and use of the Zadar Software driver for the Zadar sensors including: zSIGNAL, zPROX and zPRIME. The Zadar Software driver includes 3 main functionalities:

- **zVUE** implements the main data processing to receive the point cloud from the sensor. This is a key component required for operating zSIGNAL, zPRIME and zPROX. For zPRIME the main processing burden is implemented directly on the radar.
- **Viewer** allows users to visualize the point cloud in real-time.
- **Record** allows users to save the point cloud into CSV format.

Additionally, we provide template programs to receive the point cloud in order for you to start implementing your own custom application.



[figure 1] Sensor and Software overview for zSignal/zProx and zPrime case

2 System Requirements

2.1 System Requirements zSIGNAL / zPROX

Below are the requirements for the installation and use of the sensor with Zadar Software driver:

- Operating System
  - GNU/Linux - Tested on: x86 Ubuntu 20.04 and 22.04 LTS
    - Note: ARM Linux platforms are supported. Please contact support for non-x86 platforms.
    - Note: for other operating systems, please contact support.
  - Kernel must support SocketCAN
- Hardware
  - 1 radar sensor (zSIGNAL or zPROX)
  - 1 CAN-FD receiver
    - This can be a CAN-FD to USB interface, or it can be a CAN peripheral on your embedded system that supports 8Mbps CAN-FD through SocketCAN.
- Software
  - ZadarSoftware/ folder containing the Zadar software driver

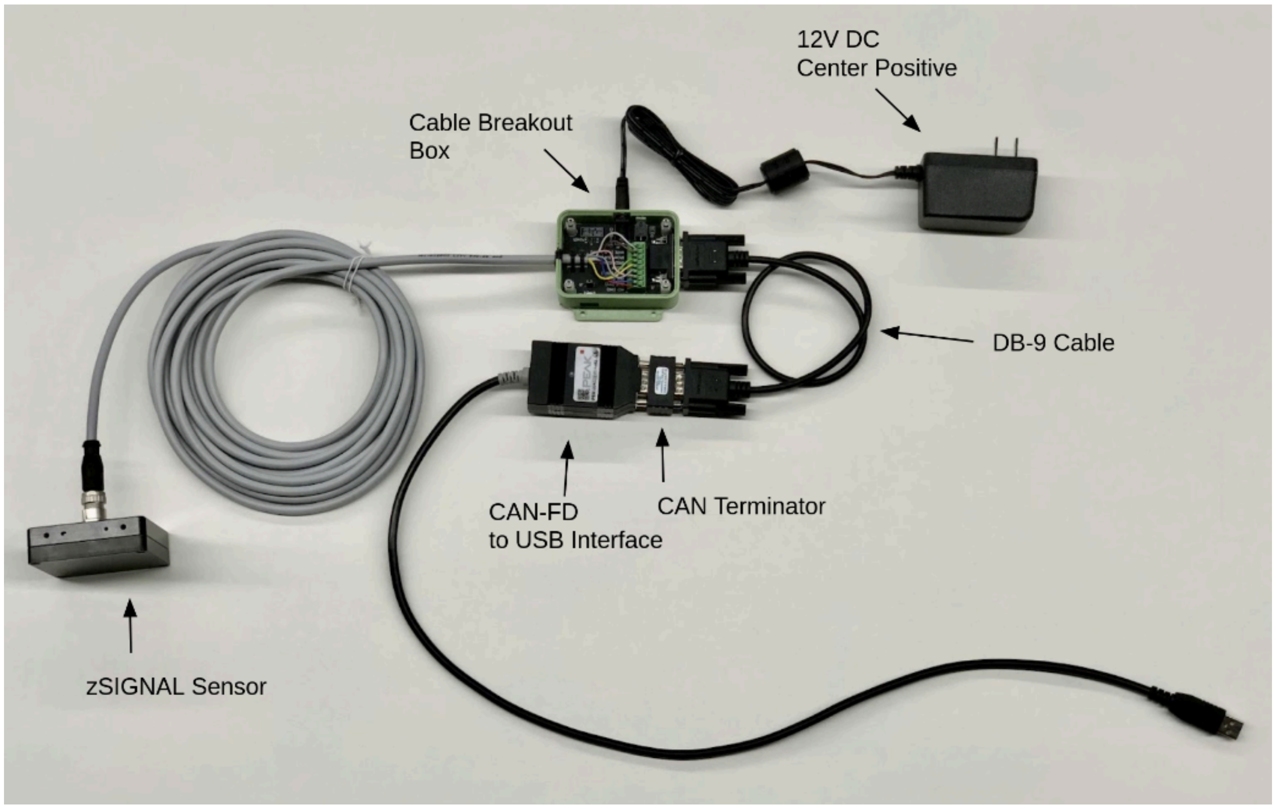
2.2 System Requirements zPRIME

Below are the requirements for the installation and use of zPrime with Zadar Software driver:

- Operating System
  - GNU/Linux - Tested on: x86 Ubuntu 20.04 and 22.04 LTS
    - Note: ARM Linux platforms are supported. Please contact support for non-x86 platforms.
    - Note: for other operating systems, please contact support.
- Hardware
  - 1 radar sensor (zPrime)
  - Power over Ethernet adapter (POE++ Power rating) **OR** 1000BASE-T to 1000BASE-T1 converter with 12V adapter
- Software
  - ZadarSoftware/ folder containing the Zadar software driver

2.3 Setup zSIGNAL / zPROX

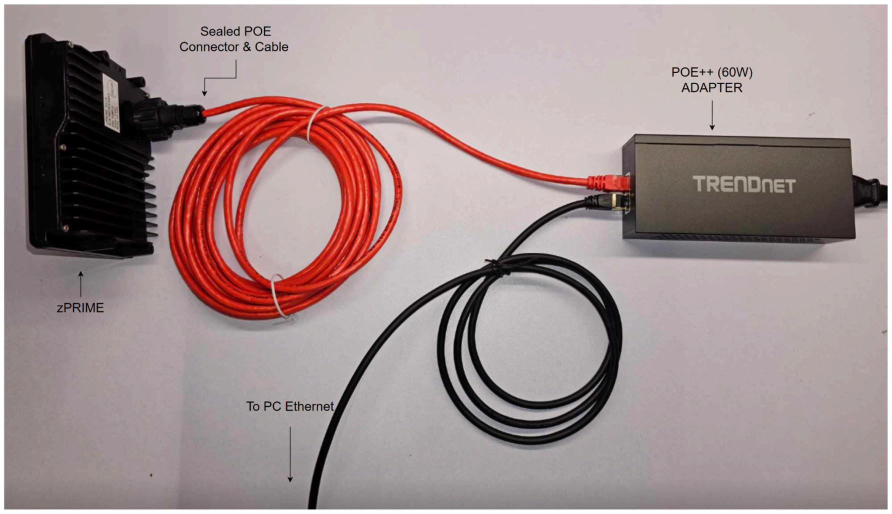
1. For your evaluation convenience, we provide a complete kit to test our sensor with any Linux computer. This includes the cable breakout, power supply, and CAN-FD interface.
2. For volume deployment in an embedded/ECU environment, zSIGNAL can integrate directly into a CAN bus with a custom harness.
3. Connect the sensor to the computer as pictured and power it up with all the required cables.
    - Sensor -> Breakout Cable -> DB-9 cable -> CAN DB9 terminator -> CAN-FD to USB interface
    - 12V 1A Power



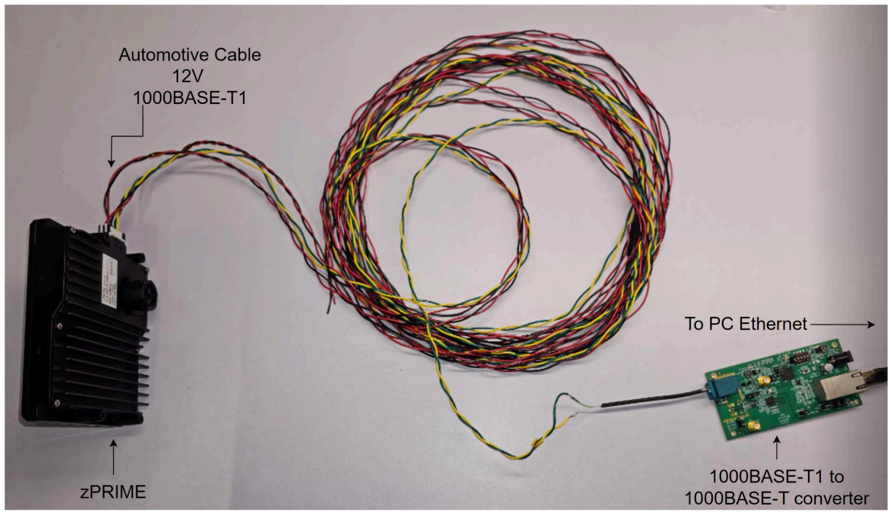
4. Extract the archive ZadarSoftware.zip, provided by our software team
5. Place yourself inside the unzipped ZadarSoftware/ folder, right-click the mouse and select “open in terminal” (<u>you will now see a window terminal where you can run the commands that will follow</u>).

2.4 Setup zPRIME

- For your evaluation convenience, we provide a complete kit to test our sensor with any Linux computer. This includes the necessary power supply and ethernet adapter.
- Connect the sensor to the computer as pictured.
    - Sensor -> POE++ Adapter -> PC (This for Automotive that already uses Ethernet)



    - Sensor -> 1000BASE-T1 to 1000BASE-T Converter -> 12V and PC *(For Automotive Requiring a 1000BASE-T1 to Ethernet Conversion)*



- Extract the archive ZadarSoftware.zip, provided by our software team
- Place yourself inside the unzipped ZadarSoftware/ folder, right-click the mouse and select "open in terminal" (you will now see a window terminal where you can run the commands that will follow).

4 Quick Start

4.1 Installing Prerequisites

Platform Prerequisites (for zSignal and zProx only)

Your system must support communication over CAN-FD. This can be achieved in two ways:

- Communication over a CAN-USB interface such as a PEAK-CAN device. This is typically used for PC/x86 hosts
- Communication over an integrated CAN-FD peripheral with 8Mbps data rate support on an embedded system. This is typically used for integrated systems such as Nvidia Jetson/Orin.

For evaluation and prototyping purposes we support testing our sensors on Linux x86 PC systems with a PEAK-CAN USB interface as pictured in the previous section. SocketCAN is used to allow our software driver to communicate with the CAN bus. To load the appropriate kernel modules:

```
$ sudo modprobe can
$ sudo modprobe peak_usb
```

For other types of interfaces (including embedded CAN peripherals), contact our support team for integration help.

Software Prerequisites

Install dependencies with the following command on the terminal:

```
$ sudo ./install_dependencies.sh
```

Alternatively, the dependencies required are:

```
 libzmq3-dev
 libgoogle-glog-dev
 libqglviewer-dev-qt5
 libserial-dev
```

Refer to the requirements.txt file for the python libraries required to run zPrime (requires python3-pip).

Wait for all dependencies to install before proceeding.

4.2 Running the Radar

With the radar powered up and connected, run the radar with the following command on the terminal. Ctrl + C will terminate the process. In case of errors, try restarting the system by power cycling the sensor and plugging/unplugging the USB-CAN interface or ethernet cable.

```
$ ./run_radar.sh <radar_num> <radar_types> <radar_sns> <mode_ids> <viewer> <log>
```

For single-radar zsignal or zprox example:

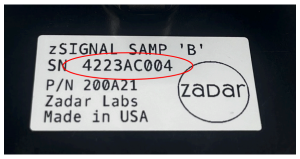
```
$ sudo ./run_radar.sh 1 zsignal 3223AC002 1 on off
```

For single-radar zprime example:

```
$ ./run_radar.sh 1 zprime 3223AC001 1 on off
```

For multi-radar pass a list of values for every argument of every radar to run for example:

```
$ sudo ./run_radar.sh 2 zsignal zsignal 3223AC001 3223AC002 1 1 on off
```

⚠	zprime must run without “sudo” privileges, while zprox and zsignal require the “sudo”.
	Replace `<radar_num>` with the correct number of sensors to launch. If more than 1 the rest of the attributes `<radar_types>`, `<radar_sns>` and `<mode_ids>` will contain a list of values for every radar respectively.
	Replace `<radar_types>` with the correct sensor type (lowercase): *zsignal*, *zprox or zprime*.
	Replace `<radar_sns>` with the correct sensor serial number you are using (label on the back of the sensor identified by 9 alphanumeric digits).  zSIGNAL SAMP 'B' SN 4223AC004 P/N 200A21 Zadar Labs Made in USA zadar
	Replace `<mode_ids>` with the correct mode number defined in the table section 4.3 for your application. If you have a custom mode defined by the Zadar team, put the correct ID for the mode as provided to you.
	Replace `<viewer>` with *on* or *off* to open the viewer if needed. More details about the viewer in section 4.4
	Replace `<log>` with *on* or *off* to save into a .txt file the terminal outputs. The file will be created into a /logs/ folder in the zadar software driver main folder.

4.3 Radar Modes

Note: custom modes may override default modes based on custom specifications.

zPROX Modes

Mode ID	1	2	3	4
Maximum Range	20m	50m	20m	50m
Minimum Range	5cm	15cm	10cm	25cm
Field of View (H x V)	140° x 120°			
Angular Resolution (HxV)	5° x 5° (dynamic)			
Range Resolution	4.4cm	11cm	9.1cm	21cm
Vehicle Detection	20m	50m	20m	50m
Pedestrian Detection	20m	>25m	20m	>25m
Doppler Ambiguity *	27m/s	27m/s	52m/s	52m/s
Doppler Resolution	0.2m/s			
Doppler Accuracy	0.05m/s			
Frame Rate	40 Hz			

*The ego speed needs to be less than 1/4 of the provided range for Doppler Ambiguity.

zSIGNAL Modes

Mode Number	1	2	3	4
Maximum Range	50m	100m	175m	175m
Minimum Range	7cm	0.5m	0.5m	1m
Field of View (H x V)	130° x 24°			
Angular Accuracy	3°			
Range Resolution	5.2cm	43cm	31.8cm	72cm
Vehicle Detection	50m	100m	175m	175m
Pedestrian Detection	50m	100m	>100m	>100m
Doppler Ambiguity *	42m/s	158m/s	85m/s	158m/s
Doppler Resolution	0.2m/s			
Doppler Accuracy	0.05m/s			
Frame Time	35ms	25ms		

*The ego speed needs to be less than 1/4 of the provided range for Doppler Ambiguity.

zPRIME Modes (Narrow / Wide FOV Variant)

Mode Number	1	2	3	4	5**	6**
Max Range	85m	85m	250m	250m	400m	400m
Minimum Range	0.1m	0.2m	0.3m	0.75m	0.5m	1m
Range Resolution	8.9cm	17.8cm	26.2cm	52.3cm	41.9cm	82.2cm
Doppler Ambiguity *	74.4m/s	143.4m/s	74.4m/s	143.4m/s	74.4m/s	143.4m/s
Doppler Resolution	0.145m/s	0.14m/s	0.145m/s	0.14m/s	0.145m/s	0.14m/s
Field of View (H x V)	130° x 24°					
Angular Resolution (H x V)	0.35° x 0.35°					
Frame Time	100ms					

* The ego speed needs to be less than 1/4 of the provided range for Doppler Ambiguity.

** Available for narrow FOV..

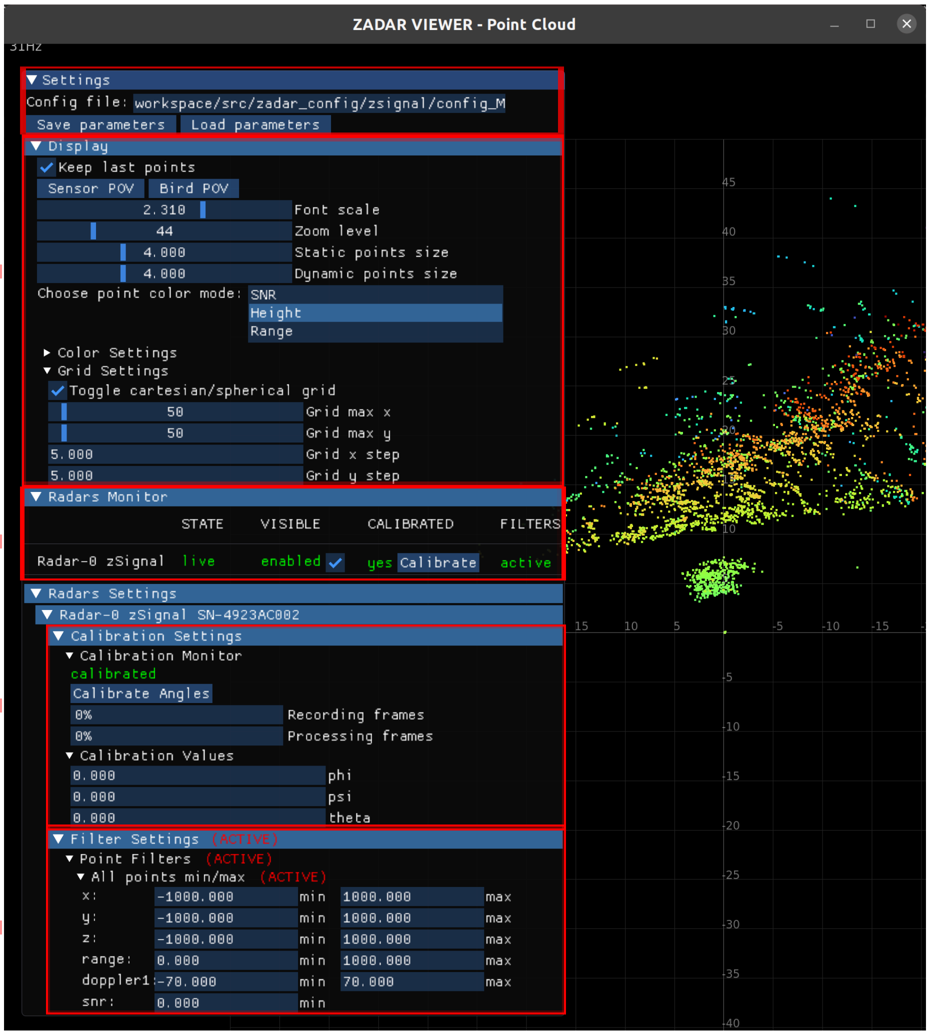
zPRIME Specification

Frequency	76-77 or 76-81GHz	
Interface	1000BASE-T1	1000BASE-T
Connections	6-pin automotive	Waterproof RJ45
Power Supply	+9-24V DC	802.3bt Type 3 (PoE++)
Power Consumption	22W avg, 34W peak based on transmit duty cycle	
Synchronization	IEEE 1588 PTP, gPTP	
Operating Temperature	-40°-85°C	
Ingress Protection	IP68 2-meter	
Dimensions	140 x 103 x 30mm	
Weight	490g	
Material	Polycarbonate black, 6061 aluminum black anodized	

4.4 Viewing the Point Cloud

We provide a point cloud viewer for quick and easy visualization of the radar's output data. The viewer window allows you to visualize the point cloud and control a few features for visualization and filtering. By holding the left mouse button you can tilt the view of the 3D point cloud, and by holding the right mouse button you can translate the view. You have a set of controls on the menu with different sections like shown in *figure 3*.

[Figure 3] Viewer window



```
ZADAR VIEWER - Point Cloud
31Hz
▼ Settings
Config file: workspace/src/zadar_config/zsignal/config_M
 Save parameters   Load parameters
▼ Display
 ✓ Keep last points
 Sensor POV   Bird POV
          2.310        Font scale
           44          Zoom level
          4.000        Static points size
          4.000        Dynamic points size
 Choose point color mode: SNR
                          Height
                          Range
 ▸ Color Settings
 ▾ Grid Settings
  ✓ Toggle cartesian/spherical grid
           50          Grid max x
           50          Grid max y
  5.000                Grid x step
  5.000                Grid y step
▼ Radars Monitor
                     STATE   VISIBLE    CALIBRATED          FILTERS
 Radar-0 zSignal     live    enabled ✓  yes Calibrate       active
▼ Radars Settings
 ▼ Radar-0 zSignal SN-4923AC002
  ▼ Calibration Settings
   ▾ Calibration Monitor
   calibrated
   Calibrate Angles
   0%                  Recording frames
   0%                  Processing frames
   ▾ Calibration Values
   0.000               phi
   0.000               psi
   0.000               theta
  ▼ Filter Settings (ACTIVE)
   ▾ Point Filters (ACTIVE)
    ▾ All points min/max (ACTIVE)
     x:        -1000.000   min   1000.000   max
     y:        -1000.000   min   1000.000   max
     z:        -1000.000   min   1000.000   max
     range:    0.000       min   1000.000   max
     doppler1: -70.000     min   70.000     max
     snr:      0.000       min
```

Save the configuration after some changes

Display settings allow to change the grid, font, zoom of the viewer and color of the points
Shortcut views:
- Bird POV: XY plane view
- Sensor POV: XZ plane view

[zSignal and zProx only] Radar Monitor provides an overview over the main status of the radar and some configurations.

[zSignal and zProx only] Calibration settings of the radar, allows the possibility to tune the position of the radar in respect to the vehicle. Also optional online automatic calibration of phi provided. Section 4.5 for more details

[zSignal and zProx only] Filter on the point cloud, applied to the x,y,z,range,doppler and snr. Filters follows a min-max logic, so only points in that range will be visible on the viewer.

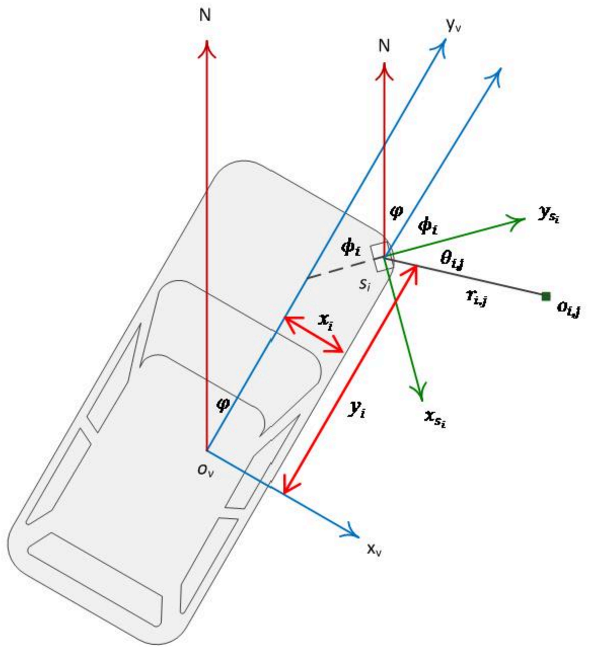
Keyboard shortcuts:
- 'a': show axes
- 'o': toggle vertical alignment
- '[space]': switches camera view from orthographic to perspective mode

4.5 Radar Calibration (zSignal and zProx only)

The angle displacement of the radar in respect of the center of mass for all 3 axes of the vehicle can either be measured precisely by the customer, or, Zadar's online calibration method can be utilized to accurately estimate φ_{rc} (azimuth phi).

The process for automatic online calibration will be as follows:

- Instruct the car driver or the ego platform (such as a robot) to proceed in a straight path without any wheel movement or rotation for at least 1 second.
- Press the calibration button on the viewer during this time (see section 4.4 for viewer calibration button)
- We recommend that the user align the theta and roll angles carefully using precise measurement devices to facilitate a smooth convergence of the algorithm.



[Figure 4] Indicative car-radar odometry

4.6 Capturing Radar Point Clouds

Record the point cloud to CSV file

While the sensor is running you have the option to record the point cloud into .csv format. By running the following command into a new terminal tab, the script will create a new folder containing the saved point cloud in csv format (1 frame per file). Ctrl+C to stop the recording.

```
$ sudo ./record_csv.sh
```

> Output format
> In the “logs/” folder you will find a new directory containing the saved files in the following format: */logs/*<timestamp>*/zadar_scan0_frame<frame_id>.csv*

Receiving Real-Time Point Clouds

In the “api_templates/” folder, we provide templates for Python and C++ scripts to allow receiving the point cloud frame by frame in real-time and connect to your own application. These templates are to be considered as a starting point to receive the point cloud and forward the data to your own pipeline, and are openly available for you to modify and customize:

- ApiSocket.py
- ApiSocket.cpp
- ApiRos.py (*Requires a specific zadar software version for ROS, please ask support from the zadar software team*)

Recording Point Clouds to a bagfile (ROS)

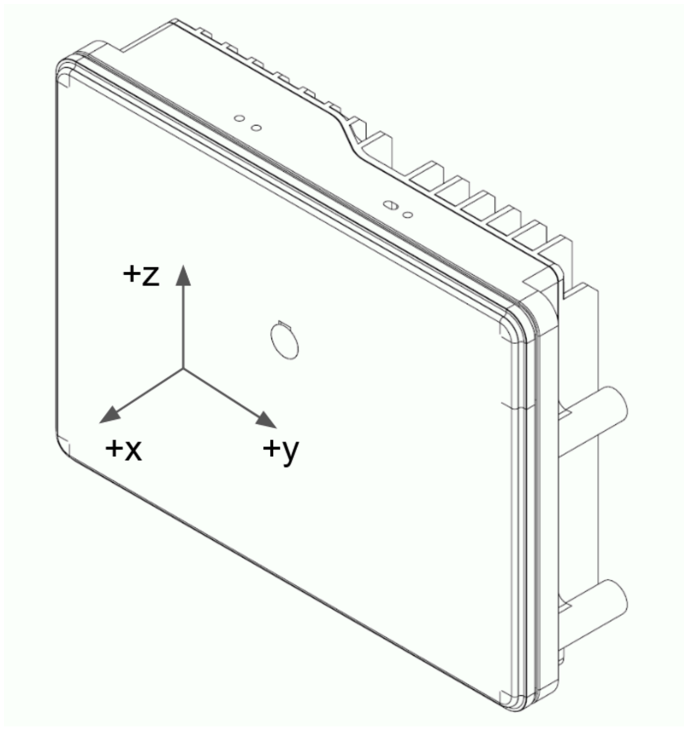
Requires a specific zadar software version for ROS, please ask support from the zadar software team

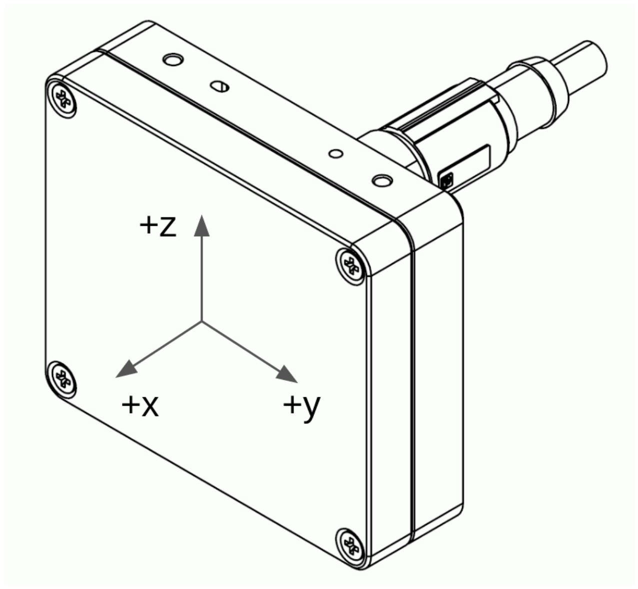
While the sensor is running, you also have the option to record the point cloud into .bag format (if you have ROS available on your system). By running the following command into a new terminal tab, the script will create a new folder containing the saved point cloud into .bag format (entire stream per file). Ctrl+C to stop the recording.
$ sudo ./record_ros.sh

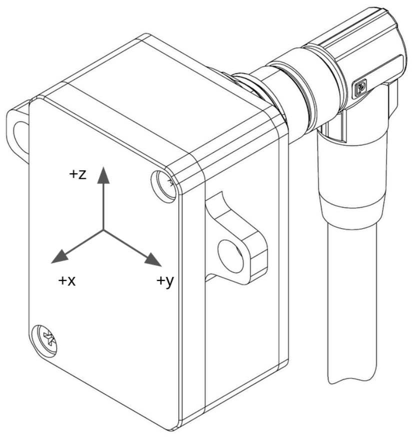
> Output format
> In the “logs/” folder you will find a new directory containing the saved files in the following format: /logs/<timestamp>.bag

5 Point Cloud Data Format

The output coordinate system per radar is as follows:







The point cloud data format of zSIGNAL/zPROX is an array of points with the following fields:

Header Per Frame

Field	Unit	Description
seq	-	Frame sequence number
timestamp	ns	Time elapsed since sensor boot
width	-	Number of points received in the frame

Data Per Point

Field	Unit	Description
x	meters	X coordinate of point in meters
y	meters	Y coordinate of point in meters
z	meters	Z coordinate of point in meters
snr	dB	Signal to noise ratio. A higher SNR means increased confidence in the received point (Only available on zSignal and zProx)
power	dB	Relative power level of the point normalized (Only available on zPrime)
range	meters	Radial range of the point in meters
noise	dB	Internal variable
doppler	meters/second	Relative radial velocity of the point as measured via Doppler shift
adjusted_doppler	meters/second	Computed absolute radial velocity of the point in the ground reference frame
frame_num	frame number	Frame number counter starting from 0
is_static	bool	Flag to indicate if a point has been detected as stationary (not moving) relative to the environment
removed	-	Internal variable (always 0)
subframe_index	-	Internal variable

6 Revision History

DATE	REVISION	NOTES
10/11/2023	2.0	Initial Release
11/06/2023	2.1	Zprox commands and modes added. Viewer layout updates
12/18/2023	2.2	Zprime and new viewer functionalities
01/12/2024	2.3	Multi-radar and new viewer functionalities
01/26/2024	2.4	Log, firmware updater and zPrime communication handler