Universal radar message proposal #5264

knzo25 started this conversation in Design

knzo25 on Sep 20 Collaborator

I would like to propose a new (and hopefully universal) radar message that overcomes the shortages and limitations of the current OSS (ROS) one.
To do do, I first would like to present an analysis of the current ROS message, then list some of the radars with OSS drivers or definitions out there, and finally propose a new set of message definitions that support all known use cases, and is expected to be compatible with future devices.

That being said, I am not expert in this area, and would hope to get feedback from the community about this proposal, so we can properly leverage radar technology to its fullest (internally at TIER IV we already went through a series of reviews regarding this proposal).

*Note: in this document the following terms are used interchangeably, since every vendor has its own definitions:

- RDI, pointcloud interface, detections, returns
- Objects, tracks, targets
- Doppler velocity, range rate

ROS radar message

https://github.com/ros-perception/radar_msgs

Here, there are two types of relevant structures:

`RadarReturn.msg` :

- range
- azimuth
- elevation
- doppler_velocity
- amplitude

`RadarTrack.msg` :

- 3d position & upper triangle covariance
- 3d velocity & upper triangle covariance
- 3d acceleration & upper triangle covariance
- 3d size & upper triangle covariance

ROS radar message limitations

`RadarReturn.msg` :

- No way to know if the radar provides real elevation or is it set to a constant (usually zero)
- Amplitude: SNR, RCS?
- Some radars provide further information. How can we accommodate said information?
    - Detection/Return vs Track/Object pairing
    - Point level classification
    - Different types of score (existence, etc)

`RadarTrack.msg`:

- NO ORIENTATION
- No easy / standard way to know if all the components of position/velocity/acceleration/sizes and their covariances are valid !

Known radar definitions

Ainstein AI

https://ainstein.ai/
https://github.com/AinsteinAI/ainstein_radar/tree/master

Detections: Id, SNR, range, range rate, azimuth, elevation
Objects: UIS, pose, twist, bounding box, value (likehood?), label

Texas Instruments mmWave radars

https://www.ti.com/sensors/mmwave-radar/overview.html
https://github.com/radar-lab/ti_mmwave_rospkg

Detections: position, range, range rate, azimuth, intensity (?), doppler bin
Sadly, no much available information

OmniPreSense

https://omnipresense.com/
https://github.com/SCU-RSL-ROS/radar_omnipresense/tree/master

Detections: range, range rate, FFT data, direction (string?!), object number

Delphi

https://www.amtechs.co.jp/product/rf/system-optical/radar/post-108.html
https://bitbucket.org/unizg-fer-lamor/radar_interface/src/master/msg/

Detections: range, range rate, azimuth, rcs, status
Objects: 2D position, 2D velocity, 2D acceleration, ID, Status, width, movable, amplitude

SmartMicro

https://www.smartmicro.com/automotive-radar
https://github.com/smartmicro/smartmicro_ros2_radars/tree/master

Detections: range, range rate, power, rcs, noise, snr, azimuth, elevation
Objects: 3d position, abolute speed, yaw, length, mileage (?), quality (?), acceleration (scalar), idle cycles, spline idx, status, class

Altos

https://www.altosradar.com/product
https://github.com/Altos-Radar/altosRadarParse/blob/main/pointCloud.h

Detections: range + variance, range range + variance, elevation, azimuth, SNR

Continental

https://www.continental-automotive.com/en/components/radars.html
https://github.com/tier4/nebula/tree/main/nebula_messages/continental_msgs/msg

ARS548 detections: range, range rate, azimuth, elevation (all with std + status flag), rcs, classification id, object id, ambiguity flag, multi target probability, positive predictive value
SRR520 detections: range, range rate, azimuth, SNR, RCS, false detection flags

ARS548 objects: ID, age (cycles), measurement status, movement status, position reference (which part of the object is being tracked), 3d position + cov, orientation + std, 3d velocity & acceleration (absolute & relative !) + std + cov, orientation + cov, existence probability, multi class classification, shape edge (length + width)
SRR520 objects: ID, 2d position + std, 2 velocity + std, 2d acceleration + std, shape edge (length + width), orientation, RCS, score, valid flag for the shape, measurement status

Towards an universal radar interface

As presented before, the ROS radar messages are insufficient since:

- Objects do not contain orientation.
- Although there is support for 4D radars, from the interface alone we can not know if the elevation is zero or is just not available.
- Different vendors and models provide several features, but there is no way to accommodate them all (mainly in the detection interface).
- We can not reliably know when a covariance is invalid, high, or simply not provided.

Notes regarding reference systems and dynamics:

- Depending on the radar, objects may be output either in the sensor frame or the base link (provided the extrinsics are given).
- Dynamics (velocity / acceleration) can be either relative or absolute (depending on the availability of motion compensation). This needs to be clear to the user or integrator

Proposal

Radar Object Info

This message is published at a low frequency and is similar to how the `CameraInfo` works for cameras. The fields in this message are radar dependent, but do NOT change over time.

Regarding the potential classes, the definition should be a superset of the classes used in object detection in Autoware. We have seen radars with classes that while important to autonomous driving, do not map with the standard classes that we use.

Availability, resolution, and range, are fields that may be needed to correctly process the information in the relevant stacks.

```
std_msgs/Header header

bool measurement_status_available #  measured or tracked
bool position_z_available
bool velocity_z_available
bool acceleration_z_available
bool length_available
bool width_available
bool height_available
bool position_cov_available
bool velocity_cov_available
bool acceleration_cov_available
bool shape_cov_available
bool orientation_available
bool orientation_std_available
bool orientation_rate_available
bool orientation_rate_std_available
bool existence_probability_available

bool position_resolution_available
bool velocity_resolution_available
bool acceleration_resolution_available
bool orientation_resolution_available
bool orientation_rate_resolution_available

bool position_range_available
bool velocity_range_available
bool acceleration_range_available
bool orientation_rate_range_available

float32 position_resolution
float32 velocity_resolution
float32 acceleration_resolution
float32 orientation_resolution
float32 orientation_rate_resolution

float32 position_max_value
float32 velocity_max_value
float32 acceleration_max_value
float32 orientation_rate_max_value

# Class definitions - not part of a specific radar but of the
interface
# ANIMAL=1 PEDESTRIAN=2 ......
# This field is required
uint32[] available_classes # The classes in this array are a
subset of the defined classes

# This field is required
bool absolute_dynamics # absolute or relative dynamics
```

Radar Objects

`RadarObjects`:

```
std_msgs/Header header
RadarObject[] objects
```

`RadarObject`:

```
uint32 object_id # unique object identifier
uint16 age # number of iterations since the object was first
detected
uint8 measurement_status #  measured o tracked

geometry_msgs/Vector3 position
geometry_msgs/Vector3 velocity
geometry_msgs/Vector3 acceleration
geometry_msgs/Vector3 shape

# Can have a predefined threshold for invalid, defined in the
message
float32[] position_cov # either 1x1 (scalar), 3x1 (diagonal), 6x1
(symmetric), or 3x3 (full)
float32[] velocity_cov # either 1x1 (scalar), 3x1 (diagonal), 6x1
(symmetric), or 3x3 (full)
float32[] acceleration_cov # either 1x1 (scalar), 3x1 (diagonal),
6x1 (symmetric), or 3x3 (full)
float32[] shape_cov # either 1x1 (scalar), 3x1 (diagonal), 6x1
(symmetric), or 3x3 (full)

float32 orientation
float32 orientation_std
float32 orientation_rate_mean
float32 orientation_rate_std

float32 existence_probability
float32[] class_probability # has the dimensionality defined in
the available_classes of the info message
```

Radar Detections Info

```
std_msgs/Header header

bool range_resolution_available
bool range_rate_resolution_available
bool elevation_resolution_available
bool azimuth_resolution_available
bool range_std_resolution_available
bool range_rate_std_resolution_available
bool elevation_std_resolution_available
bool azimuth_std_resolution_available

bool range_range_available
bool range_rate_range_available
bool elevation_range_available
bool azimuth_range_available

bool snr_range_available
bool rcs_range_available

float32 range_resolution
float32 range_rate_resolution
float32 elevation_resolution
float32 azimuth_resolution
float32 range_std_resolution
float32 range_rate_std_resolution
float32 elevation_std_resolution
float32 azimuth_std_resolution

# Min max values allow downstream tasks to know in advance blind
spots
float32 range_min_value
float32 range_max_value
float32 range_rate_max_value
float32 elevation_min_value
float32 elevation_max_value
float32 azimuth_min_value
float32 azimuth_max_value

float32 snr_min_value
float32 snr_max_value
float32 rcs_min_value
float32 rcs_max_value
```

Radar Detections

The radars detections are currently not used in Autoware. In the future, though, the data size is expected to increase and they could be integrated into sensor fusion models, at which point, an interface like `PointCloud2` would be needed.

In addition, the radar detections are the ones with the most diversity (less uniform) among different vendors. This means accommodating all vendors in a single static message is not feasible. For that reason, we propose a similar approach to `PointCloud2`:

```
std_msgs/Header header

# Similar as a pointcloud
uint32 num_detections
sensor_msgs/PointField[] fields
bool is_bigendian
uint32 point_step
uint8[] data
```

↑ 1 👀 1

3 comments · 1 reply

Oldest Newest Top

knzo25 on Sep 20 Collaborator Author

@mitsudome-r @yukkysaito @drwnz @xmfcx

↑ 1 0 replies

xmfcx on Oct 2 Maintainer

- https://github.com/ros2/common_interfaces/blob/rolling/geometry_msgs/msg/PoseWithCovariance.msg
- https://github.com/ros2/common_interfaces/blob/rolling/geometry_msgs/msg/TwistWithCovariance.msg
- https://github.com/ros2/common_interfaces/blob/rolling/geometry_msgs/msg/AccelWithCovariance.msg
- https://github.com/ros2/common_interfaces/blob/rolling/shape_msgs/msg/SolidPrimitive.msg

↑ 1 0 replies

xmfcx on Oct 2 Maintainer

```
std_msgs/Header header

bool range_resolution_available
bool range_rate_resolution_available
bool elevation_resolution_available
bool azimuth_resolution_available
bool range_std_resolution_available
bool range_rate_std_resolution_available
bool elevation_std_resolution_available
bool azimuth_std_resolution_available

bool range_range_available
bool range_rate_range_available
bool elevation_range_available
bool azimuth_range_available

bool snr_range_available
bool rcs_range_available

float32 range_resolution
float32 range_rate_resolution
float32 elevation_resolution
float32 azimuth_resolution
float32 range_std_resolution
```

```
float32 range_rate_std_resolution
float32 elevation_std_resolution
float32 azimuth_std_resolution

# Min max values allow downstream tasks to know in advance blind
spots
float32 range_min_value
float32 range_max_value
float32 range_rate_max_value
float32 elevation_min_value
float32 elevation_max_value
float32 azimuth_min_value
float32 azimuth_max_value

float32 snr_min_value
float32 snr_max_value
float32 rcs_min_value
float32 rcs_max_value
```

This message could be simplified like following:

New messages

`FloatBounds.msg`

```
# FloatBounds.msg

float32 min_value  # Minimum measurable value
float32 max_value  # Maximum measurable value
```

`MeasurementDetails.msg`

```
# MeasurementDetails.msg

float32[<=1] resolution  # Resolution of the measurement
(optional if not available)
FloatBounds[<=1] range   # Min and Max values encapsulated in
FloatBounds (optional if not available)
```

`RadarDetectionsInfo.msg`

```
# RadarDetectionsInfo.msg

std_msgs/Header header

MeasurementDetails range_info         # Information about range
measurements
MeasurementDetails range_rate_info    # Information about range
rate (Doppler velocity)
MeasurementDetails elevation_info     # Information about
elevation measurements
MeasurementDetails azimuth_info       # Information about azimuth
measurements
MeasurementDetails snr_info           # Information about Signal-
```

```
to-Noise Ratio
MeasurementDetails rcs_info          # Information about Radar
Cross Section
```

BoundedVector feature of ROS 2 messages could be used to denote availability.

@knzo25 @mojomex @drwnz what do you think?

↑ 1 1 reply

mojomex on Oct 2 Collaborator

Very clean, I like the improvement!

What would it mean for `resolution` to be unavailable? Wouldn't that make the associated measured value effectively meaningless/unusable for downstream modules?

Speaking of downstream modules, having as little exceptions/different configurations as possible would be a great help (and reduce the potential for users to make errors). Things like

- setting `resolution=1.0`, `range.min_value = -inf`, `range.max_value=inf` if not available
- for `float32[] position_cov # either 1x1 (scalar), 3x1 (diagonal), 6x1 (symmetric), or 3x3 (full)`, have the omitted values 0/duplicated from the upper triangle to have all matrices 3x3 (shouldn't be too much overhead?)

That, or have a helper library similar to PointCloudWrapper to make working with all the variations less error-prone.
What do you think @knzo25 @xmfcx @drwnz?