Automotive Rearview RADAR Tracker Documentation

Release 1.2.1

Petr Bojda

May 10, 2018

CONTENTS:

CHAPTER

ONE

INTRODUCTION

The functions described here are prepared to generate digitalized version of radio signals in both baseband and pass-band. Resuting signals are stored in form of NumPy arrays.

1.1 Dependencies:

Python 3.6.2, Numpy 1.12.

1.2 Installation

No special operation required. All what needs to be done is to save entire ‘’srcpy” folder with its contents and use any function. Once they are called from the ‘’srcpy” folder they should work immediatelly, otherwise make sure you have the ‘’srcpy” folder included in your python system path.

CHAPTER

TWO

DATA CONTAINERS MODULE

The module defines all the necessary containers to store, sort, filter, import and export data either raw radar detections or already processed tracks. Also available here are classes to keep and manipulate data from a referential DGPS system.

All the containers are based on lists of appropriate points. Lists are inherited from a python's built-in class 'list' with additional methods. Points are inherited from a basic 'object' class.

2.1 DetectionPoint

Defines a class which represents a raw radar detection.

class data_containers.**DetectionPoint**(*mcc=0*, *beam=0*, *nodet_permcc=0*, *trackID=0*, *rng=0.0*, *vel=0.0*, *azimuth=0.0*, *left=True*, *car_width=1.88*)

Creates a point as it was detected by the RADAR.

Return type object

Parameters

- **mcc** (*int*) – The state of a Master-Clock-Cycle counter of the RADAR.
- **beam** (*int*) – Identifies to which beam the detection has been assigned by the RADAR.
- **nodet_permcc** (*int*) – Number of detections for this particular MCC.
- **trackID** (*int*) – ID of the track to which the detection has been assigned if so
- **rng** (*float*) – Range of the detection measured by RADAR
- **vel** (*float*) – Velocity of the detection measured by RADAR
- **azimuth** (*float*) – Azimuth of the detection measured by RADAR
- **left** (*bool*) – TRUE if the detection was measured by the left RADAR, FALSE if by right one
- **car_width** (*float*) – The width of an EGO car. (is being used to align left and right RADAR measurement)

set_XY(*x*, *y*)

For an existing detection sets its _x and _y attributes independently

Parameters

- **x** (*float*) – value to assign to _x
- **y** (*float*) – value to assign to _y

2.2 ReferencePoint

class data_containers.**ReferencePoint**(*mccL=0*, *mccR=0*, *TAR_dist=0.0*, *TAR_distX=0.0*, *TAR_distY=0.0*, *TAR_velX=0.0*, *TAR_velY=0.0*, *TAR_hdg=0.0*, *EGO_velX=0.0*, *EGO_velY=0.0*, *EGO_accX=0.0*, *EGO_accY=0.0*, *EGO_hdg=0.0*)

Creates a point as delivered by a referential DGPS system.

Return type object

Parameters

- **mccL** (*int*) – The state of a Master-Clock-Cycle counter of the left RADAR
- **mccR** (*int*) – The state of a Master-Clock-Cycle counter of the right RADAR
- **TAR_dist** (*float*) – Target vehicle's distance from the EGO car
- **TAR_distX** (*float*) – Target vehicle's distance from the EGO car projected to the X axis
- **TAR_distY** (*float*) – Target vehicle's distance from the EGO car projected to the Y axis
- **TAR_velX** (*float*) – Target vehicle absolute velocity along the X axis
- **TAR_velY** (*float*) – Target vehicle absolute velocity along the Y axis
- **TAR_hdg** (*float*) – Target vehicle's heading
- **EGO_velX** (*float*) – EGO car's absolute velocity along the X axis
- **EGO_velY** (*float*) – EGO car's absolute velocity along the Y axis
- **EGO_accX** (*float*) – EGO car's absolute acceleration along the X axis
- **EGO_accY** (*float*) – EGO car's absolute acceleration along the Y axis
- **EGO_hdg** (*float*) – EGO car's heading

2.3 TrackPoint

class data_containers.**TrackPoint**(*mcc=0*, *beam=[]*, *x=0*, *y=0*, *dx=0*, *dy=0*, *rvelocity=0*, *razimuth=0*, *rrange=0*)

Creates a point which is a part of the track.

Return type object

Parameters

- **mcc** (*int*) – The state of a Master-Clock-Cycle counter of the RADAR.
- **beam** (*int*) – Identifies to which beam the detection has been assigned by the RADAR.
- **x** (*float*) – X coordinate
- **y** (*float*) – Y coordinate
- **dx** (*float*) – the time derivative of x
- **dy** (*float*) – the time derivative of y
- **rvelocity** (*float*) – an absolute velocity as measured by RADAR
- **razimuth** (*float*) – an azimuth as measured by RADAR

get_array()
Returns x and y coordinates and their time derivatives in a 4-element numpy array

Returns a numpy of the shape (4,1); np.array([x, dx, y, dy])

2.4 DetectionList

class data_containers.**DetectionList**

2.5 UnAssignedDetectionList

class data_containers.**UnAssignedDetectionList**(*Tsampling*, *gate*)

Parameters

- **Tsampling** –
- **gate** –

new_detection(*detection*)
Tests whether or not the list of unassigned detections can form a new track.

Parameters **detection** (DetectionPoint) – The detection which is going to be tested.

two_point_projection(*det1*, *det2*)
Extrapolates two detections in terms of the first order polynomial. The extrapolation is computed from x,y coordinates.

Parameters

- **start_detection** (DetectionPoint) – The first point of the extrapolation. X and Y coordinates are being used only.
- **end_detection** (DetectionPoint) – The second point of the extrapolation. X and Y coordinates are being used only.

Returns Projected point, X and Y coordinates are set only.

Return type *DetectionPoint*

2.6 Track

class data_containers.**Track**(*trackID*)

CHAPTER

THREE

TRACK MANAGEMENT MODULE

The module contains one single class which manages all the necessary processes to organize all tracks, starts them from the scratch and stops them accordingly. Here is being kept a list of current tracks, they are updated when a new detection is assigned to them, incoming detections are sorted out and either assigned to already existing track or they are stored into a list of unassigned detections.

3.1 TrackManager

TrackManager is a class of which primary task is to organize life of all tracks starting by their initiation to the moment when they are terminated. Advanced management as a parallel tracks merging is not implemented yet.

The main part of the TrackManager is a list of all active tracks where every track is a list itself, see *data_containers.Track()* in a file *data_containers.py* file.

Another list is a list of unassigned detections, *data_containers.UnAssignedDetectionList()* defined in a *data_containers.py* file. A newly incoming detection if not assigned to any existing track is stored here. Every newcomer is then tested if it can form a new track in combination with other detections already in a list. If not they are removed from a list and 'trashed' after couple of cycles in order to free the memory and avoid any possible false track initiation.

class track_management.**TrackManager**(*gate=None*, *tracker_type={'filter_type': 'kalman_filter', 'dim_x': 4, 'dim_z': 2}*, *Tsampling=0.05*)

append_track(*track*)

A new track is being appended to the list of tracks, a new tracking filter is also created alongside the track and is assigned to it.

Parameters **track** (Track) –

CHAPTER

FOUR

TRACKING FILTERS MODULE

The module contains two classes.

1. **EKF** - to produce an extended Kalman filter,
2. **Fading Filter** - to produce a fading memory filter.

These filters were developed based on the tutorial in .. _a link: https://pykalman.github.io/

4.1 Tracking Filter

class tracking_filters.**TrackingFilter**(*dim_x*, *dim_z*, *dim_u=0*)

Parameters

- **dim_x** –
- **dim_z** –
- **dim_u** –

4.2 Kalman Filter

class tracking_filters.**KalmanFilter**(*dim_x*, *dim_z*, *dim_u=0*)

CHAPTER

FIVE

RADAR PLOTS MODULE

The module contains set of functions which plot results from radar data pre-processing and tracking.

5.1 static_plot_grid_hist_selections

radar_plots.**static_plot_grid_hist_selections**()

5.2 static_plot_selections

radar_plots.**static_plot_selections**()

5.3 static_plotREF_selections module

radar_plots.**static_plotREF_selections**()

CHAPTER

SIX

UTILITIES PACKAGE

The package contains helpful functions which solve different linear-algebra problems.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX