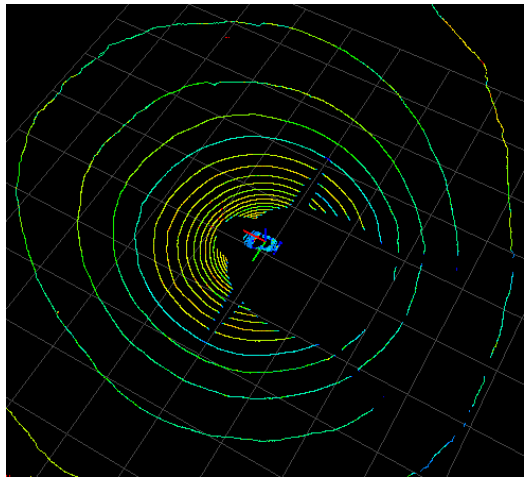
Our solution based on paper[1].

[1] "Multi-View 3D Object Detection Network for Autonomous Driving" - Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li and Tian Xia , arXiv 2016

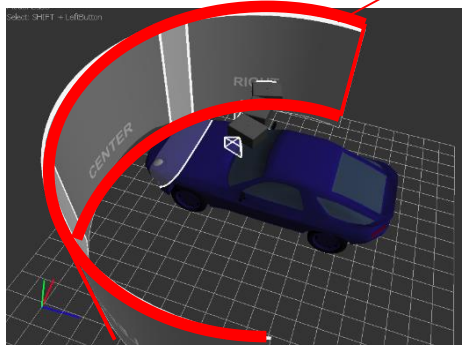
[The ideal case ...]



lidar + radar in 3d pointcloud space

top view projection

surround view projection



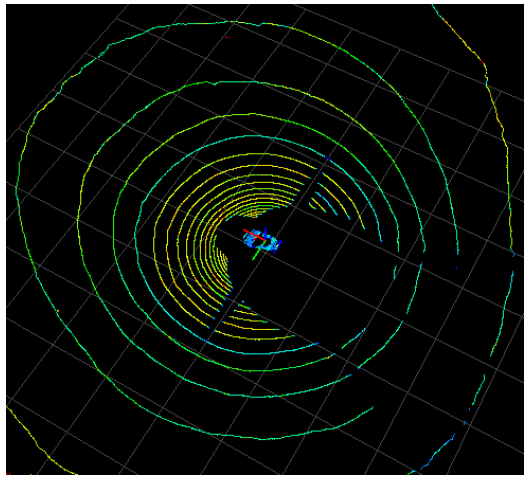
surround view projection is the full 360 degree cylindrical projection with Velodyne LiDAR at the center

camera image

end-to-end deep learning training

predicted 3d box of obstacle

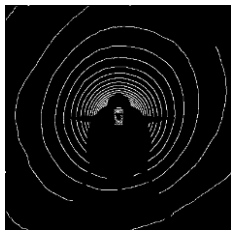
[Starter Kit]



lidar in 3d
pointcloud space

top view
projection

surround view
projection



end-to-end
deep learning
training

predicted
3d box of
obstacle

Because I have problems with time synchronisation and sensor calibration, I provided the starter kit for a simpiifed solution first. You can easily extend it include camera and radar senor data later.

[step.1 : extracting data from rosbag]

I want to do offline processing, hence i will save the images to png files and lidar point clouds to numpy npy files.
Then I will create the lidar projection images and save as npy files.
For the ground truth, I "manually annonated" one train bag to test my systems.
I use the train bag from release-2 "1/15.bag'

1. extract lidar data from rosbag.
   - my ros environemt: ubuntu 16.04, python 2.7, ros kinetic

   reference: https://discussions.udacity.com/t/installing-velodyne-drivers-on-ubuntu-16-04-lts-and-ros-kinetic/240211

   a. Install ROS Kinetic (see ros website)

   b. Install Velodyne drivers

   c. Install RViz viewer

   https://github.com/omgteam/Didi-competition-solution

   d. Download my python script to dump lidar data into python numpy npy files (saved as structured array)

   https://github.com/hengck23/didi-udacity-2017/blob/master/baseline-04/didi_data/ros_scripts/run_dump_lidar.py

e. Start the following commands in different terminals

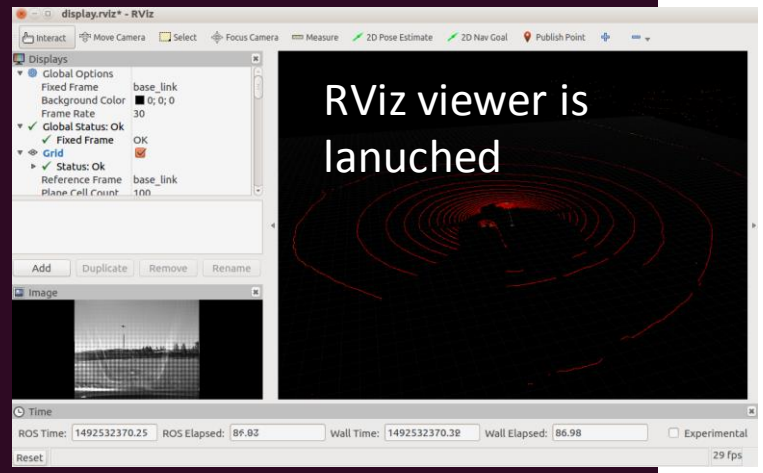
terminal1: roslaunch velodyne_pointcloud 32e_points.launch
terminal2: roslaunch didi_visualize display_rosbag_rviz.launch rosbag_file:=$BAG_FILE
terminal3: python *run_dump_lidar.py*

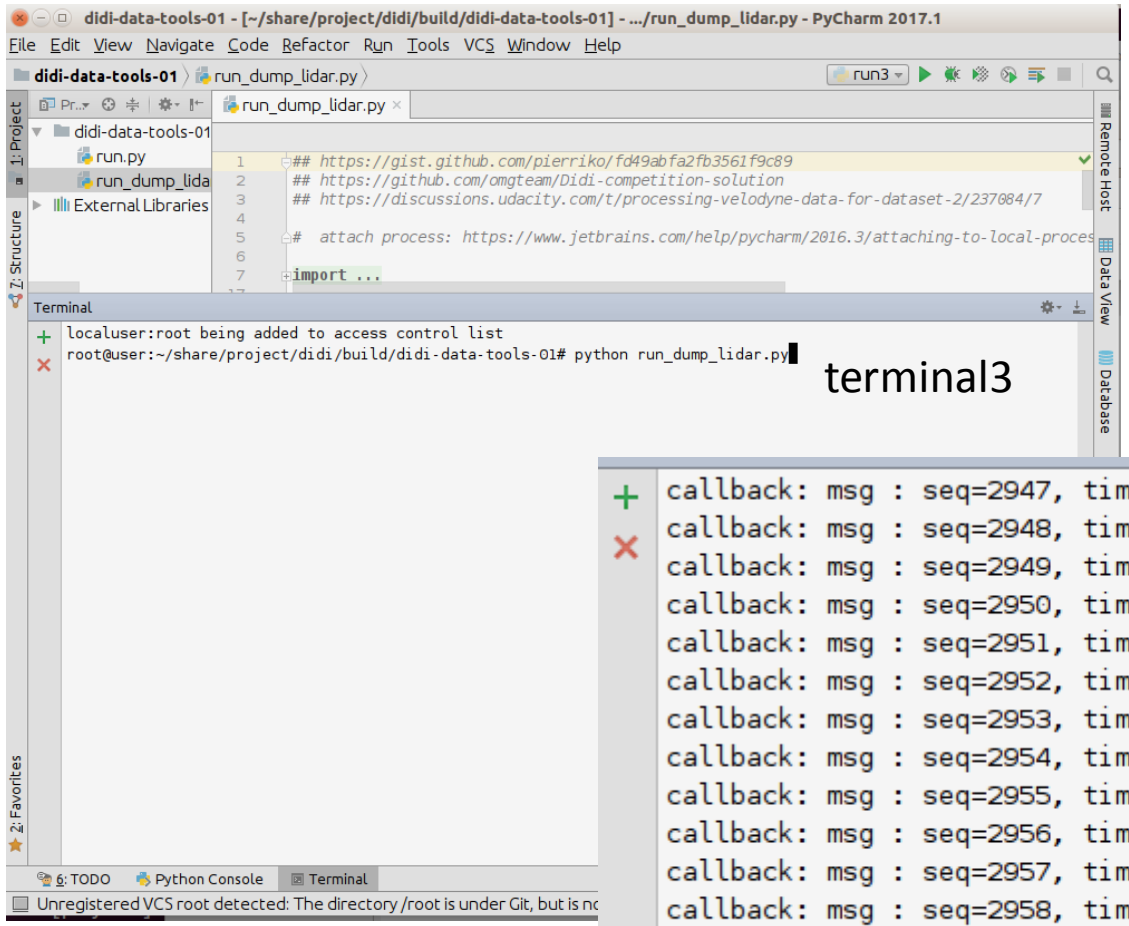
terminal1

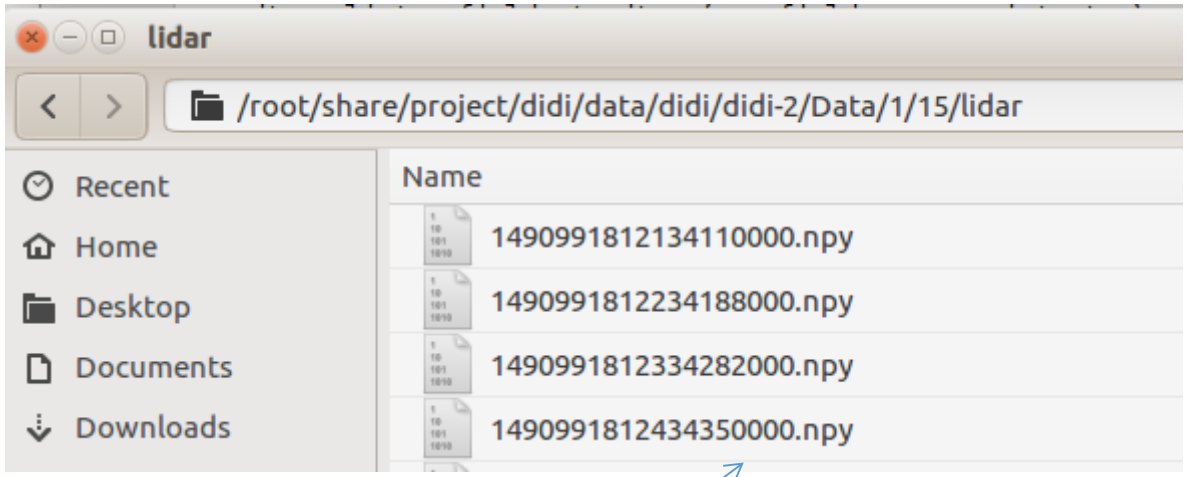
```
/opt/ros/kinetic/share/velodyne_pointcloud/launch/32e_points.launch http://localho
localuser:root being added to access control list
root@user:~# #use this to check your ip
root@user:~# hostname -I
[redacted] 172.17.0.1
root@user:~# export ROS_IP=172.17.0.1
root@user:~# export ROS_HOSTNAME=localhost
root@user:~# export ROS_MASTER_URI=http://localhost:11311
root@user:~# roslaunch velodyne_pointcloud 32e_points.launch
... logging to /root/.ros/log/3edc019e-2451-11e7-92f0-1c1b0d697254/roslaunch-
user-527.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://localhost:34405/

SUMMARY
========

PARAMETERS
 * /rosdistro: kinetic
 * /rosversion: 1.12.7
 * /velodyne_nodelet_manager_cloud/calibration: /opt/ros/kinetic/...
 * /velodyne_nodelet_manager_cloud/max_range: 130.0
 * /velodyne_nodelet_manager_cloud/min_range: 0.4
 * /velodyne_nodelet_manager_driver/device_ip:
 * /velodyne_nodelet_manager_driver/frame_id: velodyne
 * /velodyne_nodelet_manager_driver/model: 32E
 * /velodyne_nodelet_manager_driver/pcap:
 * /velodyne_nodelet_manager_driver/port: 2368
 * /velodyne_nodelet_manager_driver/read_fast: False
 * /velodyne_nodelet_manager_driver/read_once: False
 * /velodyne_nodelet_manager_driver/repeat_delay: 0.0
 * /velodyne_nodelet_manager_driver/rpm: 600.0

NODES
  /
    velodyne_nodelet_manager (nodelet/nodelet)
    velodyne_nodelet_manager_cloud (nodelet/nodelet)
    velodyne_nodelet_manager_driver (nodelet/nodelet)

auto-starting new master
process[master]: started with pid [538]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 3edc019e-2451-11e7-92f0-1c1b0d697254
process[rosout-1]: started with pid [551]
started core service [/rosout]
process[velodyne_nodelet_manager-2]: started with pid [558]
process[velodyne_nodelet_manager_driver-3]: started with pid [568]
process[velodyne_nodelet_manager_cloud-4]: started with pid [570]
```

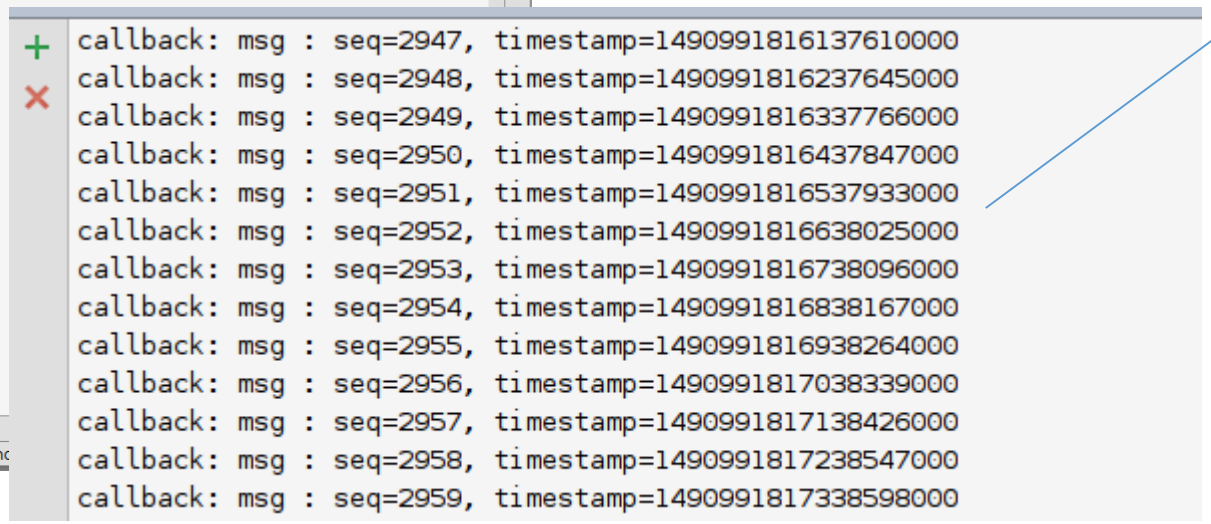
terminal2

```
root@user:~# export ROS_IP=172.17.0.1
root@user:~# export ROS_HOSTNAME=localhost
root@user:~# export ROS_MASTER_URI=http://localhost:11311
root@user:~# export CATKIN_WS=/root/share/project/didi/data/catkin_ws/kinectic/omgteam
root@user:~# export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:$CATKIN_WS/src
root@user:~# source /opt/ros/kinetic/setup.bash
root@user:~# source $CATKIN_WS/devel/setup.bash
root@user:~# export BAG_FILE=/root/share/project/didi/data/didi/didi-2/Data/1/15.bag
root@user:~# roslaunch didi_visualize display_rosbag_rviz.launch rosbag_file:=$BAG_FILE
... logging to /root/.ros/log/3edc019e-2451-11e7-92f0-1c1b0d697254/roslaunch-user-1733.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://localhost:43928/

SUMMARY
========

PARAMETERS
 * /rosdistro: kinetic
 * /rosversion: 1.12.7

NODES
  /
    link1_broadcaster (tf2_ros/static_transform_publisher)
    player (rosbag/play)
    rviz (rviz/rviz)

ROS_MASTER_URI=http://localhost:11311

core service [/rosout] found
process[player-1]: started with pid [1751]
process[link1_broadcaster-2]: started with pid [1752]
process[rviz-3]: started with pid [1753]
[ INFO] [1492532281.364529213]: Opening /root/share/project/didi/data/didi/didi-2/Data/1/15.bag

Waiting 0.2 seconds after advertising topics... done.

Hit space to toggle paused, or 's' to step.
[RUNNING]  Bag Time: 1490991818.553270   Duration: 6.635492 / 7.456196              69.70
```









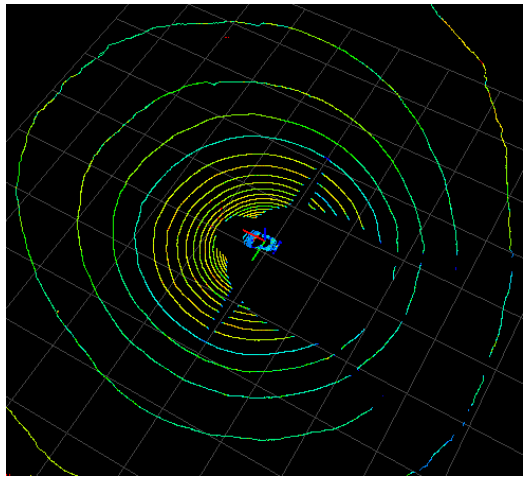
you should see it start dumping and saves the lidar pointcloud in npy

To debug code, i) lauch script using external python (becuase pycharm don't read .bashrc is is required to setup env for ROS)

ii) use pycharm 'attach to local process' for debug and stepping through breakpoints

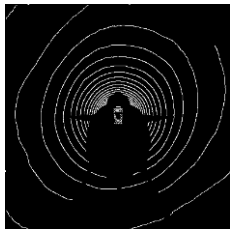
[step.2 : converting lidar dumy to projection files]

- my tensorflow renvironment: ubuntu 16.04, python 3.6, tf release 1.0



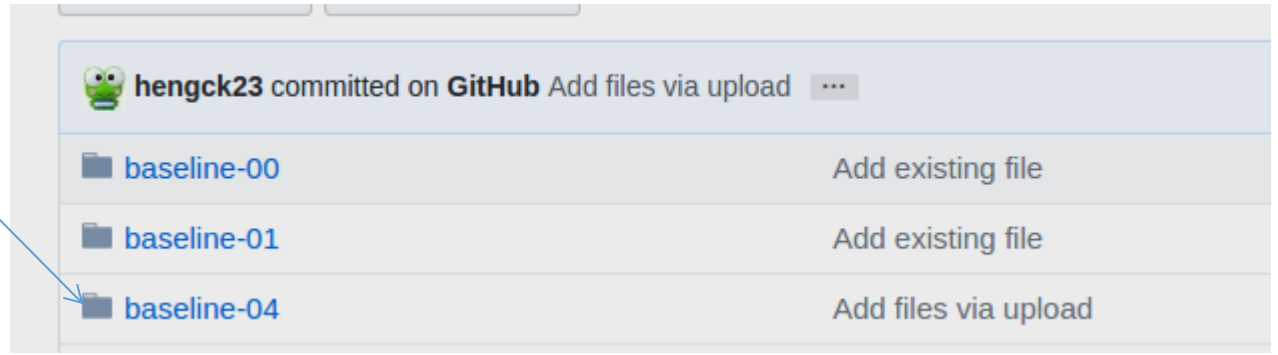
lidar in 3d
pointcloud space

→ top view
projection

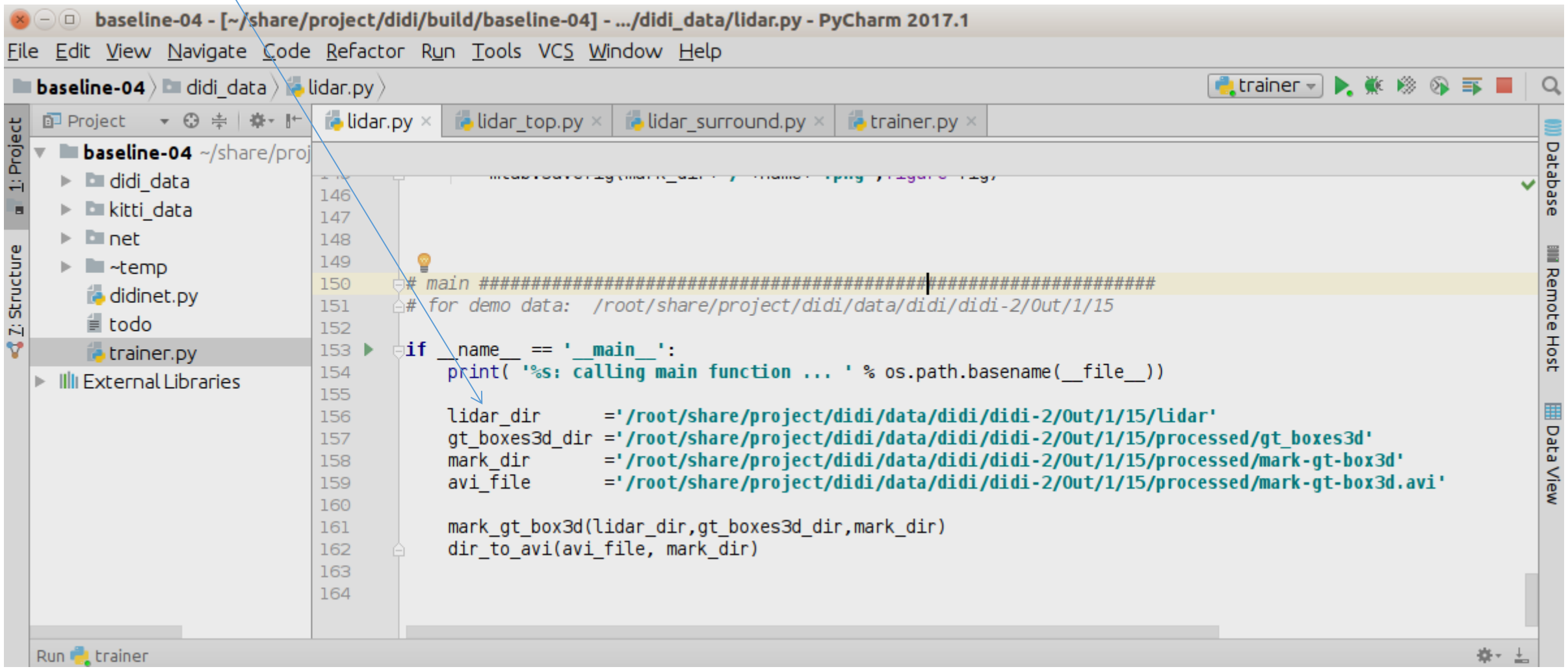


→ surround view
projection

- software version: baseline-04



use this to check dump lidar file



use this to make top view image

baseline-04 - [~/share/project/didi/build/baseline-04] - .../didi_data/lidar_top.py - PyCharm 2017.1

File Edit View Navigate Code Refactor Run Tools VCS Window Help

baseline-04 〉 didi_data 〉 lidar_top.py 〉 trainer

Project | lidar.py | lidar_top.py | lidar_surround.py | trainer.py

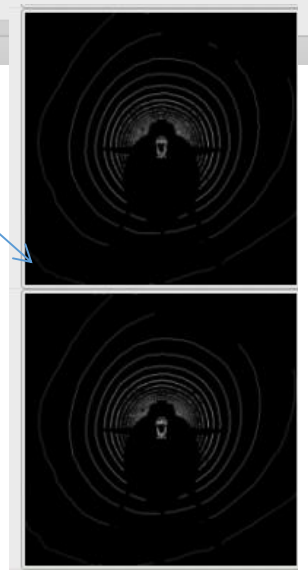
- baseline-04 ~/share/proj
  - didi_data
  - kitti_data
  - net
  - ~temp
  - didinet.py
  - todo
  - trainer.py
- External Libraries

if __name__ == … | for file in sor…

```
238
239 # main ################################################################
240 # for demo data:  /root/share/project/didi/data/didi/didi-2/Out/1/15
241
242 if __name__ == '__main__':
243
244     lidar_dir         = '/root/share/project/didi/data/didi/didi-2/Out/1/15/lidar'
245     gt_boxes3d_dir    = '/root/share/project/didi/data/didi/didi-2/Out/1/15/processed/gt_boxes3d'
246     lidar_top_dir     = '/root/share/project/didi/data/didi/didi-2/Out/1/15/processed/lidar_top'
247     lidar_top_img_dir = '/root/share/project/didi/data/didi/didi-2/Out/1/15/processed/lidar_top_img'
248     mark_dir          = '/root/share/project/didi/data/didi/didi-2/Out/1/15/processed/mark-top-box'
249     avi_file          = '/root/share/project/didi/data/didi/didi-2/Out/1/15/processed/mark-top-box.avi'
250     os.makedirs(mark_dir, exist_ok=True)
251     os.makedirs(lidar_top_dir, exist_ok=True)
252     os.makedirs(lidar_top_img_dir, exist_ok=True)
253
254     fig   = mlab.figure(figure=None, bgcolor=(0,0,0), fgcolor=None, engine=None, size=(500, 500))
255     for file in sorted(glob.glob(lidar_dir + '/*.npy')):
256         name = os.path.basename(file).replace('.npy','')
257
258         lidar_file    = lidar_dir +'/'+name+'.npy'
```

Run trainer

my results are found here



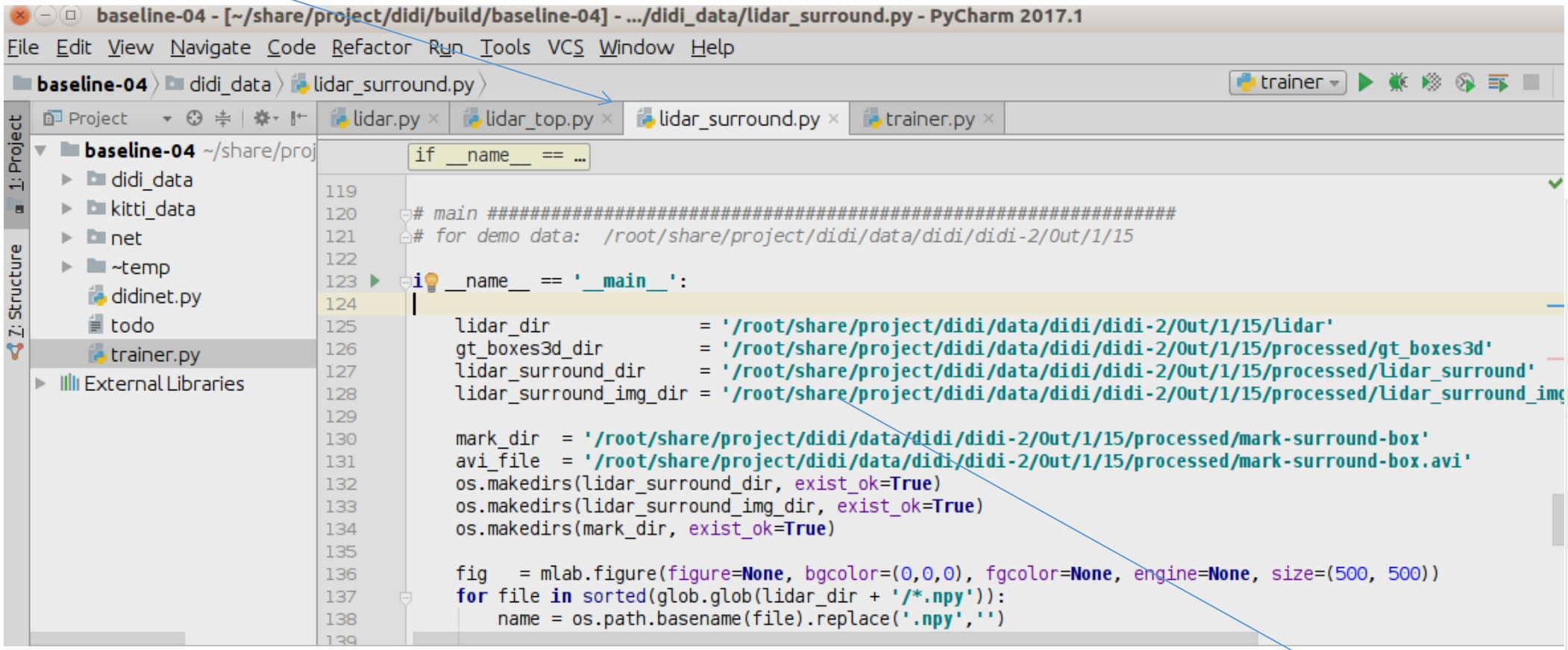
Branch: master ▾ **didi-udacity-2017** / data / didi-release-2 / 1 / 15 / **processed** /

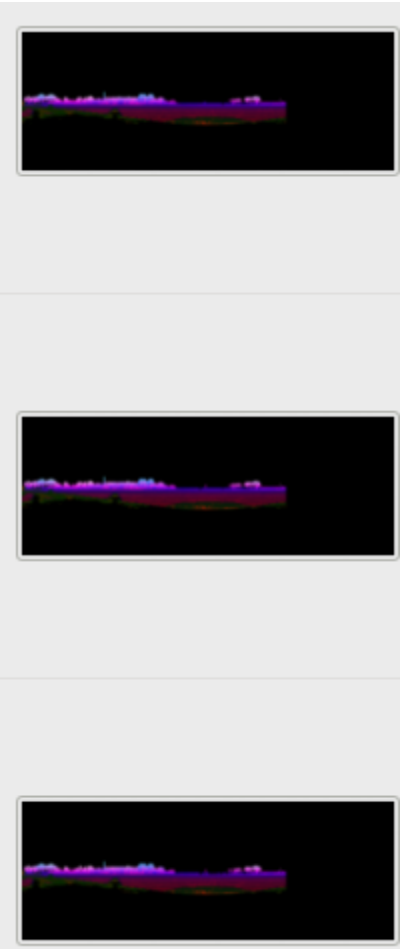
hengck23 committed on **GitHub** Add files via upload ···

..

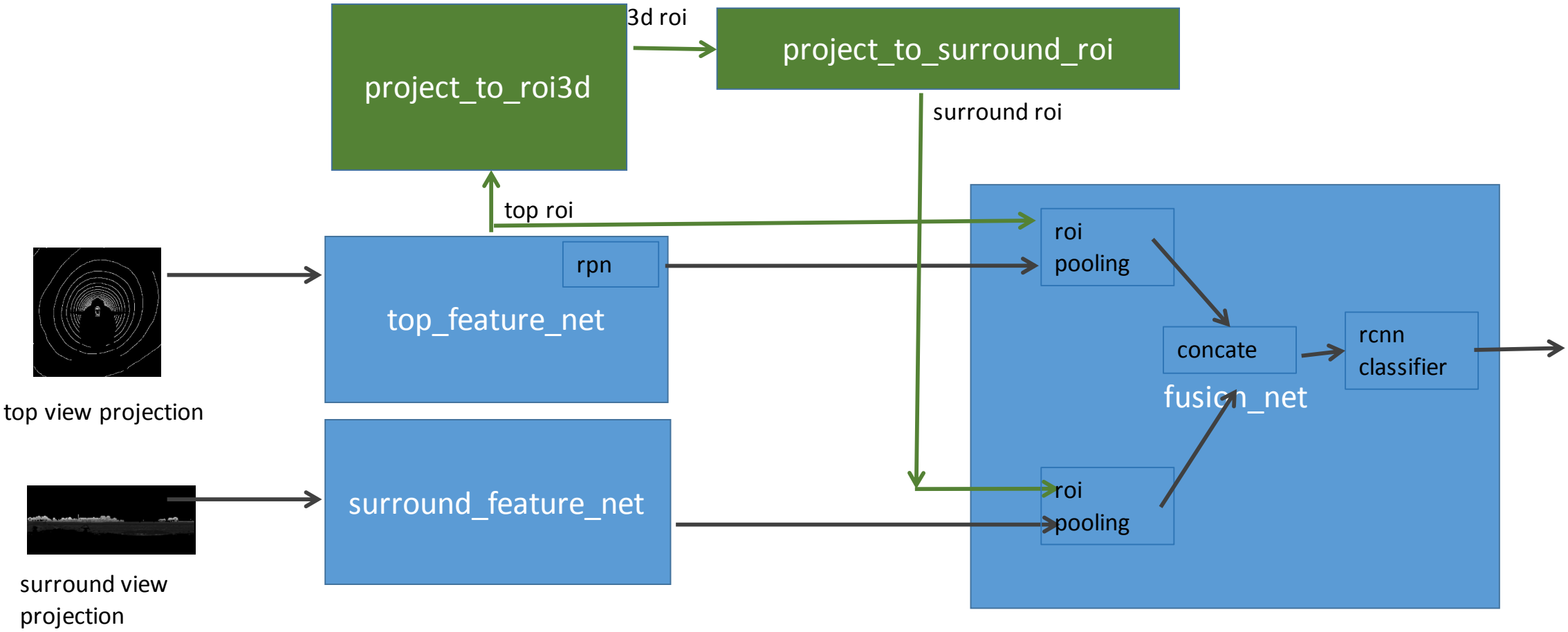
gt_boxes3d.zip	Add files via upload
lidar_surround.zip	Add files via upload
lidar_surround_img.zip	Add files via upload
lidar_top.zip	Add files via upload
lidar_top_img.zip	Add files via upload
mark-gt-box3d.avi	Add files via upload
mark-surround-box.avi	Add files via upload
mark-top-box.avi	Add files via upload
readme.txt	Create readme.txt

use this to make surround view image





[step.3 : train a simple CNN]

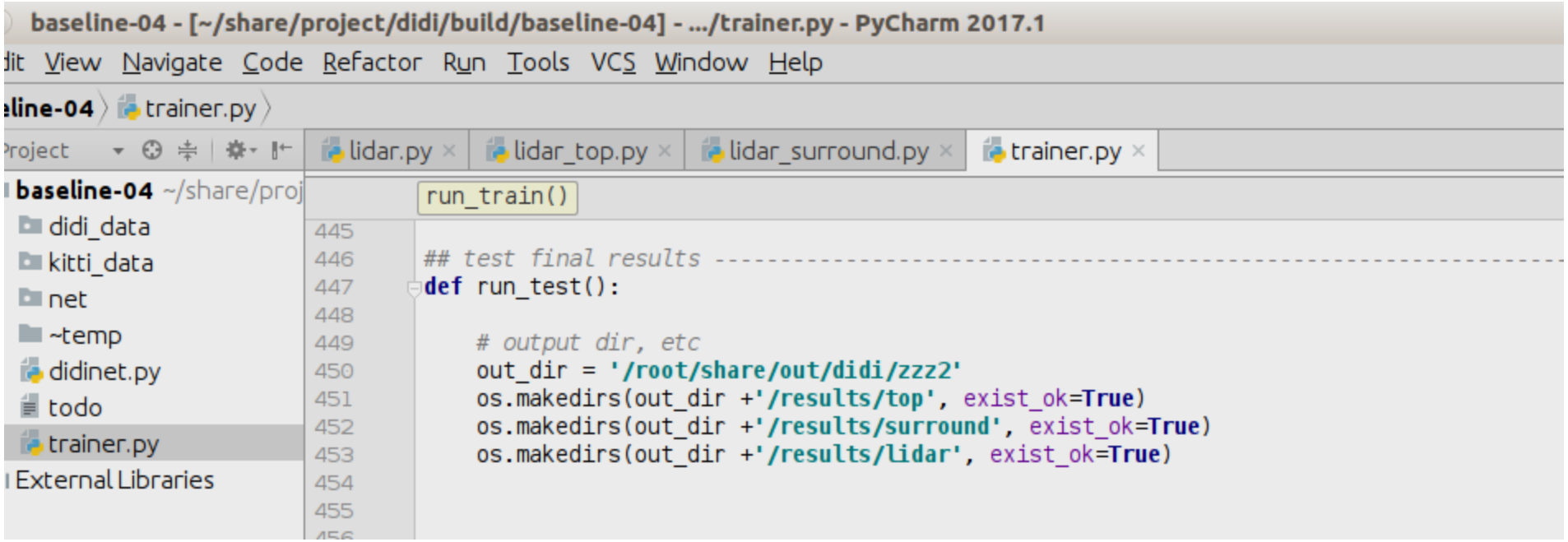


lidar.py | lidar_top.py | lidar_surround.py | trainer.py

run_train() if 1

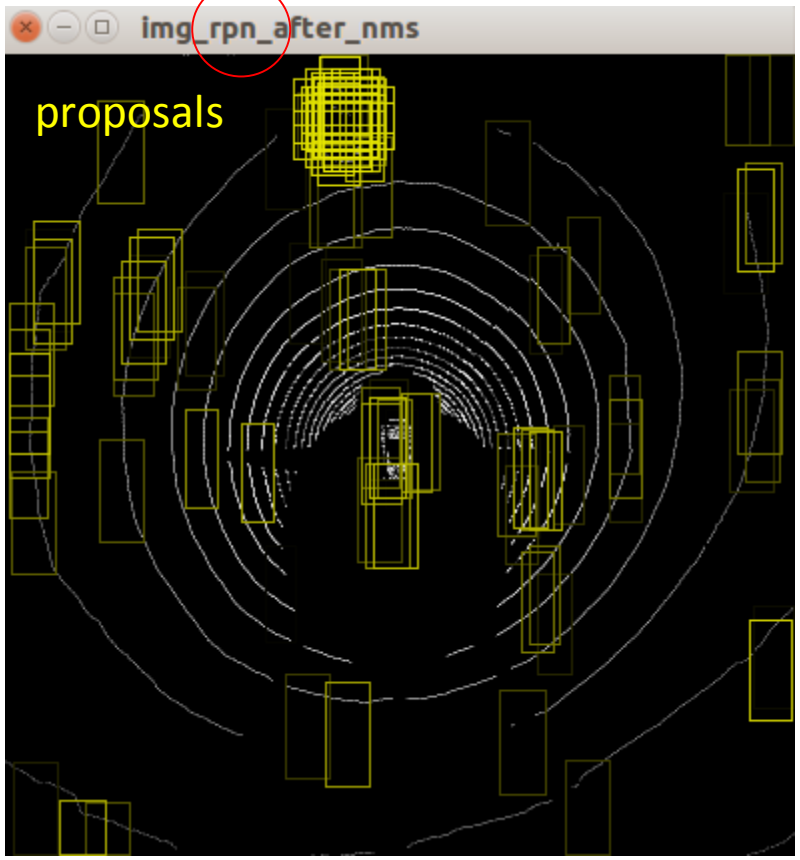
```
154
155
156
157 def run_train():
158
159     # output dir, etc
160     out_dir = '/root/share/out/didi/zzz2'
161
162     os.makedirs(out_dir +'/tf', exist_ok=True)
163     os.makedirs(out_dir +'/check_points', exist_ok=True)
164     log = Logger(out_dir+'/log.txt',mode='a')
165
166     initial_model = '/root/share/out/didi/zzz1/check_points/snap.ckpt-0'  #None for no pretrained model
167
168
```

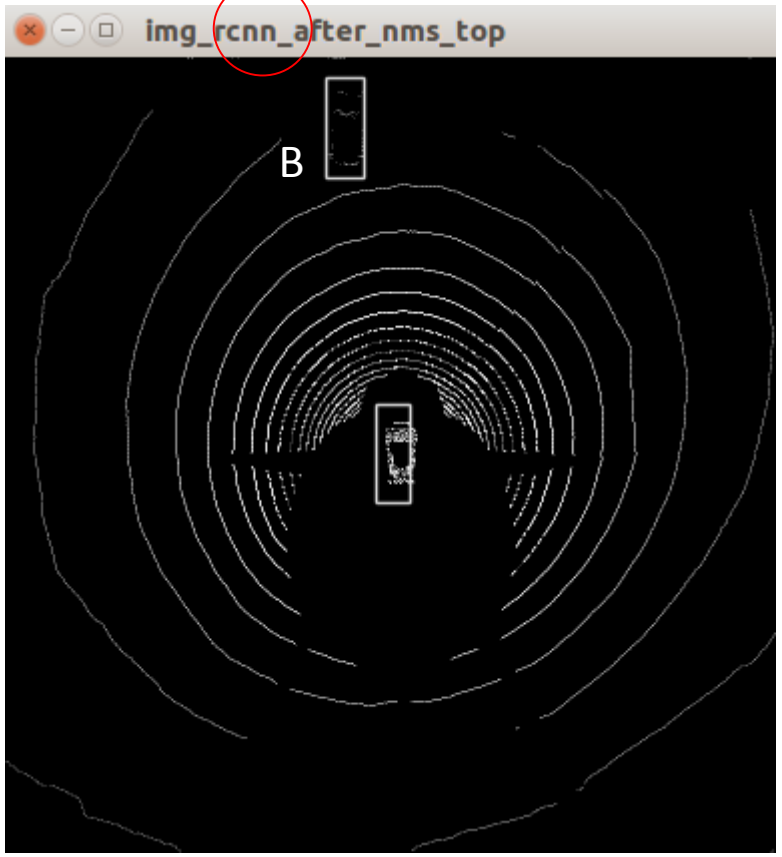
use this to train

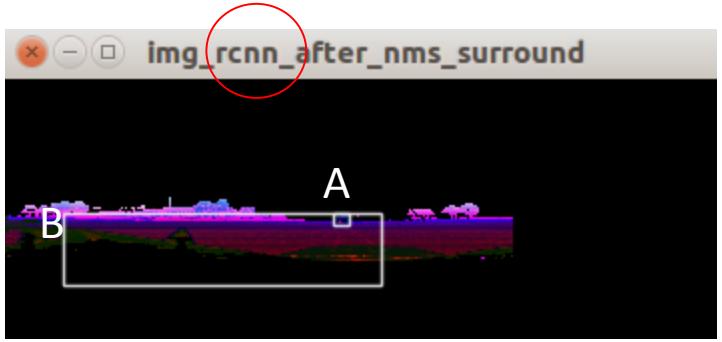


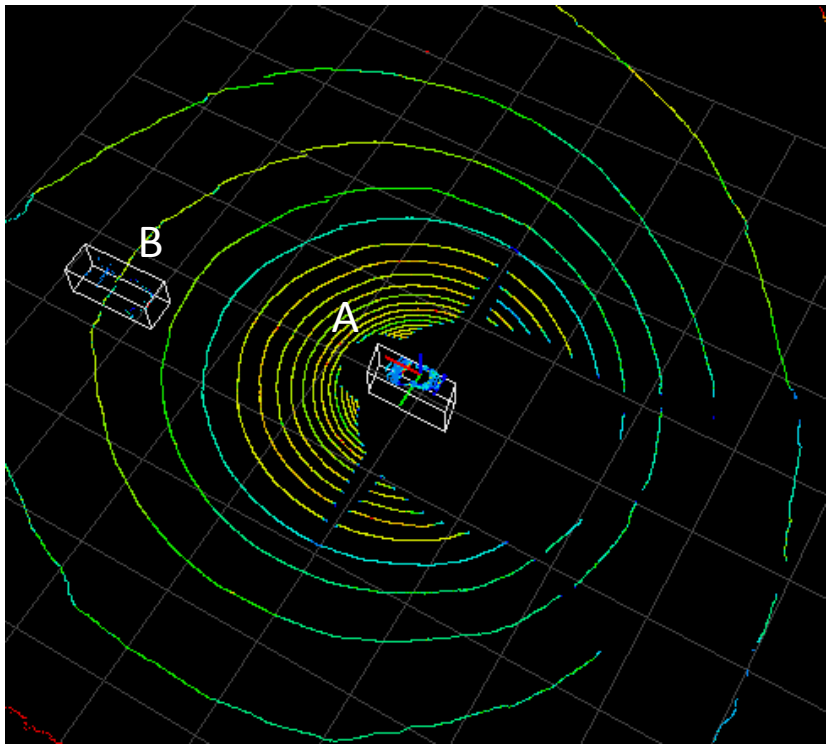
use this to test

expected results









lidar view