Behavioral Cloning

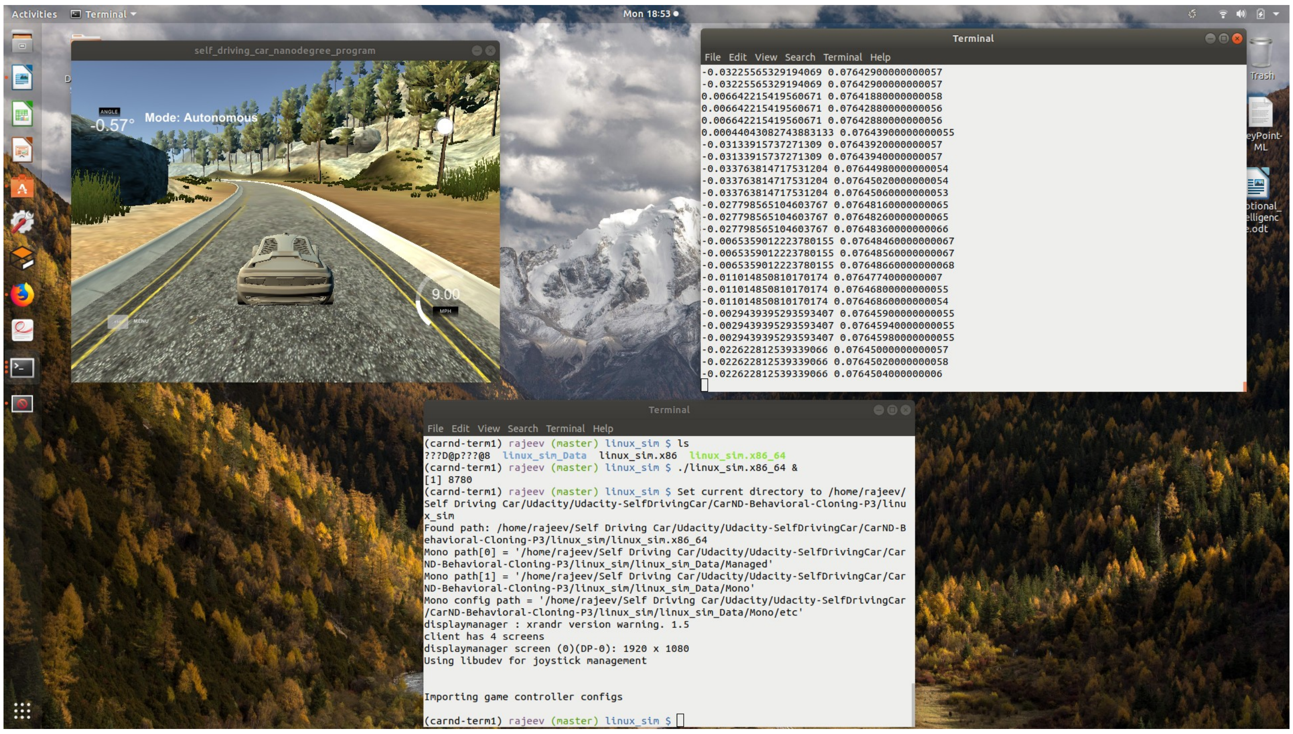
- **Behavioral Cloning Project**
  The goals / steps of this project are the following:
  a) Use the simulator to collect data of good driving behavior
  b) Build, a convolution neural network in Keras that predicts steering angles from images
  c) Train and validate the model with a training and validation set
  d) Test that the model successfully drives around track one without leaving the road
  e) Summarize the results with a written report

Image References:
Image-1 : Autonomous mode driving visualization of model.h5 through simulator
Image-2 : Nvidia CNN architecture for autonomous vehicle
Image-3 : Centre lane driving image (a sample)
Image-4 : Total number of image captured for training/validation
Image-5 : Ideal epochs for training and snap of loss convergence

- **Files Submitted & Code Quality**
  Submission includes all required files and can be used to run the simulator in autonomous mode, my project includes the following files:
  1. model.py : containing the script to create and train the model
  2. drive.py : for driving the car in autonomous mode
  3. model.h5 : containing a trained convolution neural network
  4. video.mp4 : containing a video of autonomous vehicle drive by trained model
  5. writeup_report.md or writeup_report.pdf summarizing the results

- **Submission includes functional code**
  Using the Udacity provided simulator and my drive.py file, the car can be driven autonomously around the track by executing - "sh python drive.py model.h5", also recorded video outcome as guided in class work.

Image-1 : Autonomous mode driving visualization of model.h5 through simulator



```
-0.03225565329194069 0.07642900000000057
-0.03225565329194069 0.07642900000000057
0.006642215419560671 0.07641880000000058
0.006642215419560671 0.07642880000000056
0.006642215419560671 0.07642880000000056
0.00044043082743883133 0.07643900000000055
-0.03133915737271309 0.07643920000000057
-0.03133915737271309 0.07643940000000057
-0.033763814717531204 0.07644980000000054
-0.033763814717531204 0.07645020000000054
-0.033763814717531204 0.07645060000000053
-0.027798565104603767 0.07648160000000065
-0.027798565104603767 0.07648260000000065
-0.027798565104603767 0.07648360000000066
-0.0065359012223780155 0.07648460000000067
-0.0065359012223780155 0.07648560000000067
-0.0065359012223780155 0.07648660000000068
-0.011014850810170174 0.0764774000000007
-0.011014850810170174 0.07646800000000055
-0.011014850810170174 0.07646860000000054
-0.0029439395293593407 0.07645900000000055
-0.0029439395293593407 0.07645940000000055
-0.0029439395293593407 0.07645980000000055
-0.022622812539339066 0.07645000000000057
-0.022622812539339066 0.07645020000000058
-0.022622812539339066 0.0764504000000006
```

```
(carnd-term1) rajeev (master) linux_sim $ ls
???D@p???@8  linux_sim_Data  linux_sim.x86  linux_sim.x86_64
(carnd-term1) rajeev (master) linux_sim $ ./linux_sim.x86_64 &
[1] 8780
(carnd-term1) rajeev (master) linux_sim $ Set current directory to /home/rajeev/
Self Driving Car/Udacity/Udacity-SelfDrivingCar/CarND-Behavioral-Cloning-P3/linu
x_sim
Found path: /home/rajeev/Self Driving Car/Udacity/Udacity-SelfDrivingCar/CarND-B
ehavioral-Cloning-P3/linux_sim/linux_sim.x86_64
Mono path[0] = '/home/rajeev/Self Driving Car/Udacity/Udacity-SelfDrivingCar/Car
ND-Behavioral-Cloning-P3/linux_sim/linux_sim_Data/Managed'
Mono path[1] = '/home/rajeev/Self Driving Car/Udacity/Udacity-SelfDrivingCar/Car
ND-Behavioral-Cloning-P3/linux_sim/linux_sim_Data/Mono'
Mono config path = '/home/rajeev/Self Driving Car/Udacity/Udacity-SelfDrivingCar
/CarND-Behavioral-Cloning-P3/linux_sim/linux_sim_Data/Mono/etc'
displaymanager : xrandr version warning. 1.5
client has 4 screens
displaymanager screen (0)(DP-0): 1920 x 1080
Using libudev for joystick management


Importing game controller configs

(carnd-term1) rajeev (master) linux_sim $ 
```

- **Submission code is usable and readable**

The model.py file contains the code for training and saving the convolution neural network (model.h5). The file shows the pipeline i used for training and validating the model, and it contains elobrative comments to explain how the code works. Code has modularity, independent functions and only system defined parameters (to avoid hardcodings).

Model Architecture and Training Strategy

- **Solution Design Approach**

The overall strategy for deriving a model architecture was to first try out very simple architecture (one convolution & connected layer) as explained in curriculam to understand impact of various parameters thereafter try out complex model LeNet and Nvidia model architecture.

While tried very simple architecture and unprocessed input data, trained model was able to drive vehicle for a short distance then derailed from track in any direction. While introduced pre-processing mechanism as follows, performance improved a bit:

  - Grayscale - RGB to BGR images processing (model.py code line : 52)
  - Normalization – i.e. Lambda normalization and cropping (model.py code line : 64 – 68)

However trained models couldn’t drive vehicle efficiently so decided to move on slightly complex model i.e LeNet architecture (model.py code line : 71 - 81). Outcome improved slightly but model was not able to drive vehicle on sharp turns, leaving track and went in undesired space.

Eventually as suggested in course lectures to achieve project objectives, I decided to move on Nvidia architecture model and implemented same (model.py code line : 84 - 96). This model worked well on whole track though at a few occassions go in extreme left/right, also once strike with roadside object.

Appropriate training data

- Training data was chosen to keep the vehicle driving on the road. I used a combination of center lane driving, recovering from the left and right sides of the road.
- For details about how I created the training data, see the next section (model.py code line : 47)
- Than decide to increase training data (reverse track data input for training) and fine tunning of hyper parameters.

At the end of the process with Nvidia model, vehicle is able to drive autonomously around the track without leaving the road.

Model parameter tuning

- The model used an adam optimizer, so the learning rate was not tuned manually (model.py line 100).
- Increase learning iteration to 10 (model.py code line : 33) because training and validation losss was dropping.

Attempt to reduce overfitting in the model

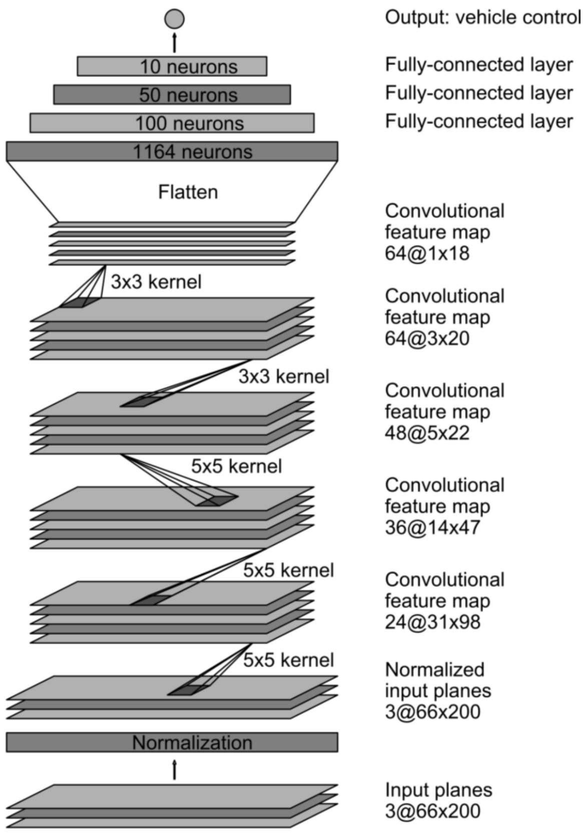
- The model was trained and validated on different data sets to ensure that the model was not overfitting (model.py code line 34), 25% data kept for validation.
- The model was tested by running it through the simulator and ensuring that the vehicle could stay on the track.

➢ **Final Model Architecture**

The final model architecture (model.py lines 84 -96) consisted of a convolution neural network with the following layers and layer sizes :

- Image-1 visualize architecture of Nvidia CNN model adopted by nvidia autonomous vehicle team. Same advance model we realized in our project to achieve expected objectives. Model has 5 convolutional layers, 3 fully connected layers, one pre-processing layer along with input/output layers.
- Five 2D convolutional layers have sizes:
  - 24, 5, 5, subsample=(2,2), activation="relu"
  - 36, 5, 5, subsample=(2,2), activation="relu"
  - 48, 5, 5, subsample=(2,2), activation="relu"
  - 64, 3, 3, activation="relu"
  - 64, 3, 3, activation="relu"

Image-2 : Nvidia CNN architecture for autonomous vehicle



➢ **Creation of the Training Set & Training Process**

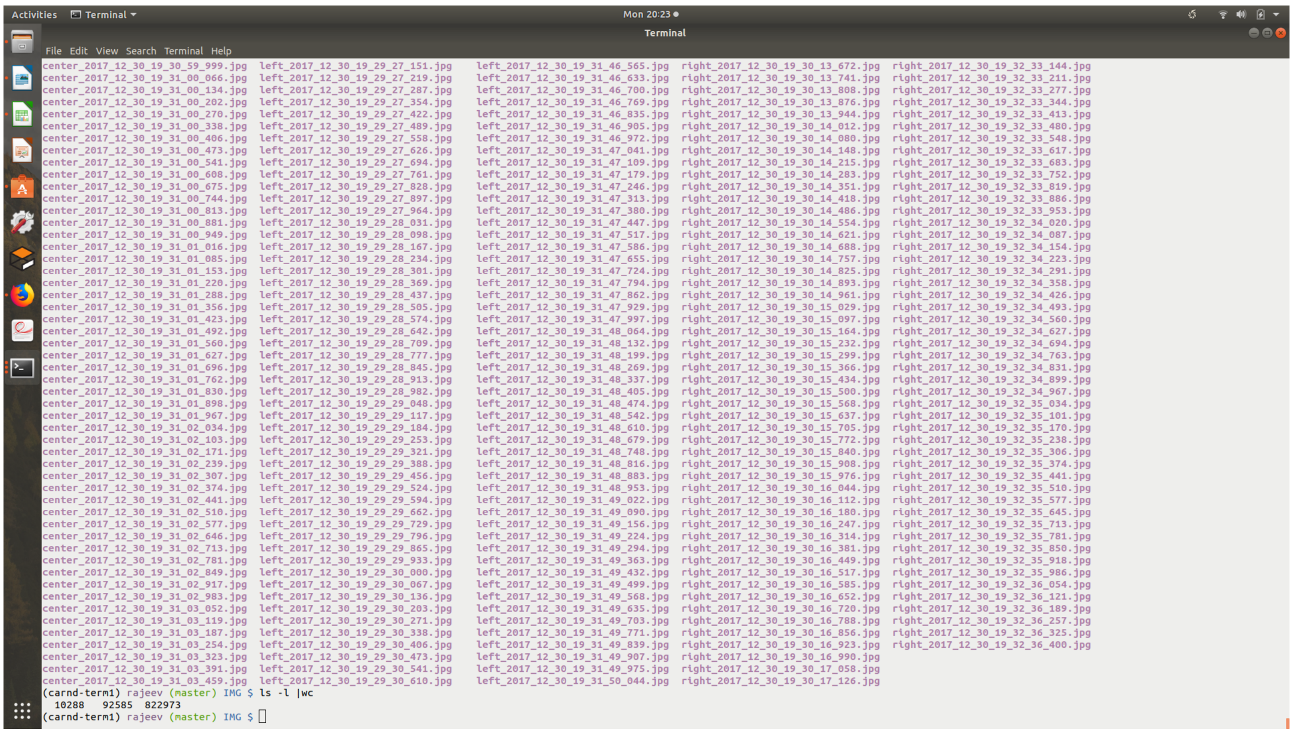
To capture good driving behavior,

- I first recorded one laps on track one using center lane driving
- Thereafter one lap on track in reverse direction
- Randomly shuffled the data set and put 25% of the data into a validation set.
- I used an adam optimizer so that manually training the learning rate wasn't necessary
- 3439 vehicle training sample data point gathered (at unique time instances)
- Date pre-processing done (explained earlier)

Here is an example image of center lane driving:

Image-3 : Centre lane driving image (a sample)

After collection process, I had ***<u>3439</u>*** number of data points for each camera (left, centre and right – file driving_log.csv columns). The total number of camera images = ***<u>10288</u>*** , as depicted in image-4

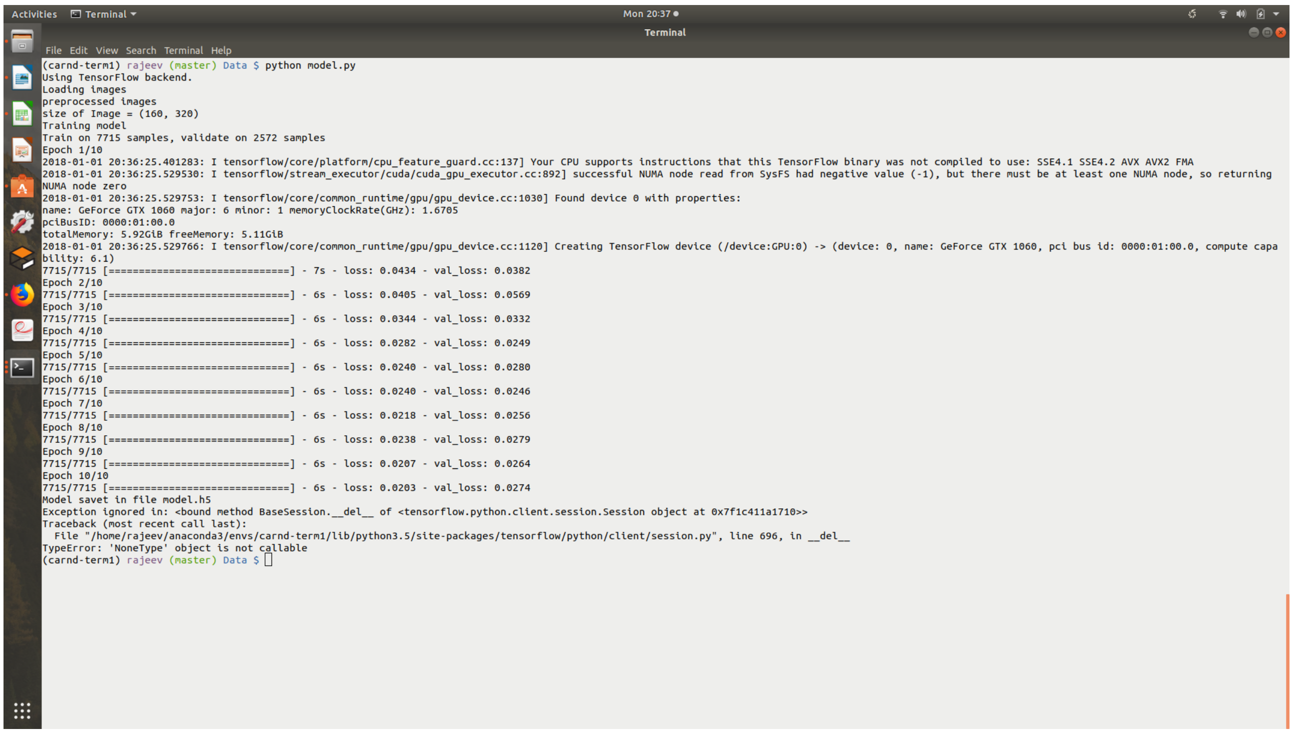


```
center_2017_12_30_19_30_59_999.jpg  left_2017_12_30_19_29_27_151.jpg    left_2017_12_30_19_31_46_565.jpg  right_2017_12_30_19_30_13_672.jpg  right_2017_12_30_19_32_33_144.jpg
center_2017_12_30_19_31_00_066.jpg  left_2017_12_30_19_29_27_219.jpg    left_2017_12_30_19_31_46_633.jpg  right_2017_12_30_19_30_13_741.jpg  right_2017_12_30_19_32_33_211.jpg
center_2017_12_30_19_31_00_134.jpg  left_2017_12_30_19_29_27_287.jpg    left_2017_12_30_19_31_46_700.jpg  right_2017_12_30_19_30_13_808.jpg  right_2017_12_30_19_32_33_277.jpg
center_2017_12_30_19_31_00_202.jpg  left_2017_12_30_19_29_27_354.jpg    left_2017_12_30_19_31_46_769.jpg  right_2017_12_30_19_30_13_876.jpg  right_2017_12_30_19_32_33_344.jpg
center_2017_12_30_19_31_00_270.jpg  left_2017_12_30_19_29_27_422.jpg    left_2017_12_30_19_31_46_835.jpg  right_2017_12_30_19_30_13_944.jpg  right_2017_12_30_19_32_33_413.jpg
center_2017_12_30_19_31_00_338.jpg  left_2017_12_30_19_29_27_489.jpg    left_2017_12_30_19_31_46_905.jpg  right_2017_12_30_19_30_14_012.jpg  right_2017_12_30_19_32_33_480.jpg
center_2017_12_30_19_31_00_406.jpg  left_2017_12_30_19_29_27_558.jpg    left_2017_12_30_19_31_46_972.jpg  right_2017_12_30_19_30_14_080.jpg  right_2017_12_30_19_32_33_548.jpg
center_2017_12_30_19_31_00_473.jpg  left_2017_12_30_19_29_27_626.jpg    left_2017_12_30_19_31_47_041.jpg  right_2017_12_30_19_30_14_148.jpg  right_2017_12_30_19_32_33_617.jpg
center_2017_12_30_19_31_00_541.jpg  left_2017_12_30_19_29_27_694.jpg    left_2017_12_30_19_31_47_109.jpg  right_2017_12_30_19_30_14_215.jpg  right_2017_12_30_19_32_33_683.jpg
center_2017_12_30_19_31_00_608.jpg  left_2017_12_30_19_29_27_761.jpg    left_2017_12_30_19_31_47_179.jpg  right_2017_12_30_19_30_14_283.jpg  right_2017_12_30_19_32_33_752.jpg
center_2017_12_30_19_31_00_675.jpg  left_2017_12_30_19_29_27_828.jpg    left_2017_12_30_19_31_47_246.jpg  right_2017_12_30_19_30_14_351.jpg  right_2017_12_30_19_32_33_819.jpg
center_2017_12_30_19_31_00_744.jpg  left_2017_12_30_19_29_27_897.jpg    left_2017_12_30_19_31_47_313.jpg  right_2017_12_30_19_30_14_418.jpg  right_2017_12_30_19_32_33_886.jpg
center_2017_12_30_19_31_00_813.jpg  left_2017_12_30_19_29_27_964.jpg    left_2017_12_30_19_31_47_380.jpg  right_2017_12_30_19_30_14_486.jpg  right_2017_12_30_19_32_33_953.jpg
center_2017_12_30_19_31_00_881.jpg  left_2017_12_30_19_29_28_031.jpg    left_2017_12_30_19_31_47_447.jpg  right_2017_12_30_19_30_14_554.jpg  right_2017_12_30_19_32_34_020.jpg
center_2017_12_30_19_31_00_949.jpg  left_2017_12_30_19_29_28_098.jpg    left_2017_12_30_19_31_47_517.jpg  right_2017_12_30_19_30_14_621.jpg  right_2017_12_30_19_32_34_087.jpg
center_2017_12_30_19_31_01_016.jpg  left_2017_12_30_19_29_28_167.jpg    left_2017_12_30_19_31_47_586.jpg  right_2017_12_30_19_30_14_688.jpg  right_2017_12_30_19_32_34_154.jpg
center_2017_12_30_19_31_01_085.jpg  left_2017_12_30_19_29_28_234.jpg    left_2017_12_30_19_31_47_655.jpg  right_2017_12_30_19_30_14_757.jpg  right_2017_12_30_19_32_34_223.jpg
center_2017_12_30_19_31_01_153.jpg  left_2017_12_30_19_29_28_301.jpg    left_2017_12_30_19_31_47_724.jpg  right_2017_12_30_19_30_14_825.jpg  right_2017_12_30_19_32_34_291.jpg
center_2017_12_30_19_31_01_220.jpg  left_2017_12_30_19_29_28_369.jpg    left_2017_12_30_19_31_47_794.jpg  right_2017_12_30_19_30_14_893.jpg  right_2017_12_30_19_32_34_358.jpg
center_2017_12_30_19_31_01_288.jpg  left_2017_12_30_19_29_28_437.jpg    left_2017_12_30_19_31_47_862.jpg  right_2017_12_30_19_30_14_961.jpg  right_2017_12_30_19_32_34_426.jpg
center_2017_12_30_19_31_01_356.jpg  left_2017_12_30_19_29_28_505.jpg    left_2017_12_30_19_31_47_929.jpg  right_2017_12_30_19_30_15_029.jpg  right_2017_12_30_19_32_34_493.jpg
center_2017_12_30_19_31_01_423.jpg  left_2017_12_30_19_29_28_574.jpg    left_2017_12_30_19_31_47_997.jpg  right_2017_12_30_19_30_15_097.jpg  right_2017_12_30_19_32_34_560.jpg
center_2017_12_30_19_31_01_492.jpg  left_2017_12_30_19_29_28_642.jpg    left_2017_12_30_19_31_48_064.jpg  right_2017_12_30_19_30_15_164.jpg  right_2017_12_30_19_32_34_627.jpg
center_2017_12_30_19_31_01_560.jpg  left_2017_12_30_19_29_28_709.jpg    left_2017_12_30_19_31_48_132.jpg  right_2017_12_30_19_30_15_232.jpg  right_2017_12_30_19_32_34_694.jpg
center_2017_12_30_19_31_01_627.jpg  left_2017_12_30_19_29_28_777.jpg    left_2017_12_30_19_31_48_199.jpg  right_2017_12_30_19_30_15_299.jpg  right_2017_12_30_19_32_34_763.jpg
center_2017_12_30_19_31_01_696.jpg  left_2017_12_30_19_29_28_845.jpg    left_2017_12_30_19_31_48_269.jpg  right_2017_12_30_19_30_15_366.jpg  right_2017_12_30_19_32_34_831.jpg
center_2017_12_30_19_31_01_762.jpg  left_2017_12_30_19_29_28_913.jpg    left_2017_12_30_19_31_48_337.jpg  right_2017_12_30_19_30_15_434.jpg  right_2017_12_30_19_32_34_899.jpg
center_2017_12_30_19_31_01_830.jpg  left_2017_12_30_19_29_28_982.jpg    left_2017_12_30_19_31_48_405.jpg  right_2017_12_30_19_30_15_500.jpg  right_2017_12_30_19_32_34_967.jpg
center_2017_12_30_19_31_01_898.jpg  left_2017_12_30_19_29_29_048.jpg    left_2017_12_30_19_31_48_474.jpg  right_2017_12_30_19_30_15_568.jpg  right_2017_12_30_19_32_35_034.jpg
center_2017_12_30_19_31_01_967.jpg  left_2017_12_30_19_29_29_117.jpg    left_2017_12_30_19_31_48_542.jpg  right_2017_12_30_19_30_15_637.jpg  right_2017_12_30_19_32_35_101.jpg
center_2017_12_30_19_31_02_034.jpg  left_2017_12_30_19_29_29_184.jpg    left_2017_12_30_19_31_48_610.jpg  right_2017_12_30_19_30_15_705.jpg  right_2017_12_30_19_32_35_170.jpg
center_2017_12_30_19_31_02_103.jpg  left_2017_12_30_19_29_29_253.jpg    left_2017_12_30_19_31_48_679.jpg  right_2017_12_30_19_30_15_772.jpg  right_2017_12_30_19_32_35_238.jpg
center_2017_12_30_19_31_02_171.jpg  left_2017_12_30_19_29_29_321.jpg    left_2017_12_30_19_31_48_748.jpg  right_2017_12_30_19_30_15_840.jpg  right_2017_12_30_19_32_35_306.jpg
center_2017_12_30_19_31_02_239.jpg  left_2017_12_30_19_29_29_388.jpg    left_2017_12_30_19_31_48_816.jpg  right_2017_12_30_19_30_15_908.jpg  right_2017_12_30_19_32_35_374.jpg
center_2017_12_30_19_31_02_307.jpg  left_2017_12_30_19_29_29_456.jpg    left_2017_12_30_19_31_48_883.jpg  right_2017_12_30_19_30_15_976.jpg  right_2017_12_30_19_32_35_441.jpg
center_2017_12_30_19_31_02_374.jpg  left_2017_12_30_19_29_29_524.jpg    left_2017_12_30_19_31_48_953.jpg  right_2017_12_30_19_30_16_044.jpg  right_2017_12_30_19_32_35_510.jpg
center_2017_12_30_19_31_02_441.jpg  left_2017_12_30_19_29_29_594.jpg    left_2017_12_30_19_31_49_022.jpg  right_2017_12_30_19_30_16_112.jpg  right_2017_12_30_19_32_35_577.jpg
center_2017_12_30_19_31_02_510.jpg  left_2017_12_30_19_29_29_662.jpg    left_2017_12_30_19_31_49_090.jpg  right_2017_12_30_19_30_16_180.jpg  right_2017_12_30_19_32_35_645.jpg
center_2017_12_30_19_31_02_577.jpg  left_2017_12_30_19_29_29_729.jpg    left_2017_12_30_19_31_49_156.jpg  right_2017_12_30_19_30_16_247.jpg  right_2017_12_30_19_32_35_713.jpg
center_2017_12_30_19_31_02_646.jpg  left_2017_12_30_19_29_29_796.jpg    left_2017_12_30_19_31_49_224.jpg  right_2017_12_30_19_30_16_314.jpg  right_2017_12_30_19_32_35_781.jpg
center_2017_12_30_19_31_02_713.jpg  left_2017_12_30_19_29_29_865.jpg    left_2017_12_30_19_31_49_294.jpg  right_2017_12_30_19_30_16_381.jpg  right_2017_12_30_19_32_35_850.jpg
center_2017_12_30_19_31_02_781.jpg  left_2017_12_30_19_29_29_933.jpg    left_2017_12_30_19_31_49_363.jpg  right_2017_12_30_19_30_16_449.jpg  right_2017_12_30_19_32_35_918.jpg
center_2017_12_30_19_31_02_849.jpg  left_2017_12_30_19_29_30_000.jpg    left_2017_12_30_19_31_49_432.jpg  right_2017_12_30_19_30_16_517.jpg  right_2017_12_30_19_32_35_986.jpg
center_2017_12_30_19_31_02_917.jpg  left_2017_12_30_19_29_30_067.jpg    left_2017_12_30_19_31_49_499.jpg  right_2017_12_30_19_30_16_585.jpg  right_2017_12_30_19_32_36_054.jpg
center_2017_12_30_19_31_02_983.jpg  left_2017_12_30_19_29_30_136.jpg    left_2017_12_30_19_31_49_568.jpg  right_2017_12_30_19_30_16_652.jpg  right_2017_12_30_19_32_36_121.jpg
center_2017_12_30_19_31_03_052.jpg  left_2017_12_30_19_29_30_203.jpg    left_2017_12_30_19_31_49_635.jpg  right_2017_12_30_19_30_16_720.jpg  right_2017_12_30_19_32_36_189.jpg
center_2017_12_30_19_31_03_119.jpg  left_2017_12_30_19_29_30_271.jpg    left_2017_12_30_19_31_49_703.jpg  right_2017_12_30_19_30_16_788.jpg  right_2017_12_30_19_32_36_257.jpg
center_2017_12_30_19_31_03_187.jpg  left_2017_12_30_19_29_30_338.jpg    left_2017_12_30_19_31_49_771.jpg  right_2017_12_30_19_30_16_856.jpg  right_2017_12_30_19_32_36_325.jpg
center_2017_12_30_19_31_03_254.jpg  left_2017_12_30_19_29_30_406.jpg    left_2017_12_30_19_31_49_839.jpg  right_2017_12_30_19_30_16_923.jpg  right_2017_12_30_19_32_36_400.jpg
center_2017_12_30_19_31_03_323.jpg  left_2017_12_30_19_29_30_473.jpg    left_2017_12_30_19_31_49_907.jpg  right_2017_12_30_19_30_16_990.jpg
center_2017_12_30_19_31_03_391.jpg  left_2017_12_30_19_29_30_541.jpg    left_2017_12_30_19_31_49_975.jpg  right_2017_12_30_19_30_17_058.jpg
center_2017_12_30_19_31_03_459.jpg  left_2017_12_30_19_29_30_610.jpg    left_2017_12_30_19_31_50_044.jpg  right_2017_12_30_19_30_17_126.jpg
(carnd-term1) rajeev (master) IMG $ ls -l |wc
  10288   92585  822973
(carnd-term1) rajeev (master) IMG $ 
```

Image-4 : Total number of image captured for training/validation

I used this training data for training the model. The validation set helped determine if the model was over or under fitting.

The ideal number of epochs was 10 as evidenced by image-5 (Outcome slightly varied for each training)



```
(carnd-term1) rajeev (master) Data $ python model.py
Using TensorFlow backend.
Loading images
preprocessed images
size of Image = (160, 320)
Training model
Train on 7715 samples, validate on 2572 samples
Epoch 1/10
2018-01-01 20:36:25.401283: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
2018-01-01 20:36:25.529530: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:892] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning
NUMA node zero
2018-01-01 20:36:25.529753: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1030] Found device 0 with properties:
name: GeForce GTX 1060 major: 6 minor: 1 memoryClockRate(GHz): 1.6705
pciBusID: 0000:01:00.0
totalMemory: 5.92GiB freeMemory: 5.11GiB
2018-01-01 20:36:25.529766: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:0) -> (device: 0, name: GeForce GTX 1060, pci bus id: 0000:01:00.0, compute capa
bility: 6.1)
7715/7715 [==============================] - 7s - loss: 0.0434 - val_loss: 0.0382
Epoch 2/10
7715/7715 [==============================] - 6s - loss: 0.0405 - val_loss: 0.0569
Epoch 3/10
7715/7715 [==============================] - 6s - loss: 0.0344 - val_loss: 0.0332
Epoch 4/10
7715/7715 [==============================] - 6s - loss: 0.0282 - val_loss: 0.0249
Epoch 5/10
7715/7715 [==============================] - 6s - loss: 0.0240 - val_loss: 0.0280
Epoch 6/10
7715/7715 [==============================] - 6s - loss: 0.0240 - val_loss: 0.0246
Epoch 7/10
7715/7715 [==============================] - 6s - loss: 0.0218 - val_loss: 0.0256
Epoch 8/10
7715/7715 [==============================] - 6s - loss: 0.0238 - val_loss: 0.0279
Epoch 9/10
7715/7715 [==============================] - 6s - loss: 0.0207 - val_loss: 0.0264
Epoch 10/10
7715/7715 [==============================] - 6s - loss: 0.0203 - val_loss: 0.0274
Model savet in file model.h5
Exception ignored in: <bound method BaseSession.__del__ of <tensorflow.python.client.session.Session object at 0x7f1c411a1710>>
Traceback (most recent call last):
  File "/home/rajeev/anaconda3/envs/carnd-term1/lib/python3.5/site-packages/tensorflow/python/client/session.py", line 696, in __del__
TypeError: 'NoneType' object is not callable
(carnd-term1) rajeev (master) Data $
```

Image-5 : Ideal epochs for training and snap of loss convergence

Summary:

Model.py, captured training set and above mentioned tunned hyper parameters able to generate a model.h5 outcome which efficiently drive autonomous vehicle. Mostly vehicle remain in centre of the road, never being erratic in driving and speed remained nearly 10mph. The video outcome is available with submission for review. All execution done Ubuntu 17.10, GPU (GeForce GTX 1060)/CPU grade machine and compute execution being fairly fast.