Behavioral cloning

Objective:

The goals is to teach the car how to behave with the road through steering angle by learning a lot of images with steering angles

Input: images for the road from right, left and center cameras mounted on the car

Output: steering angles

Data Set Summary & Exploration

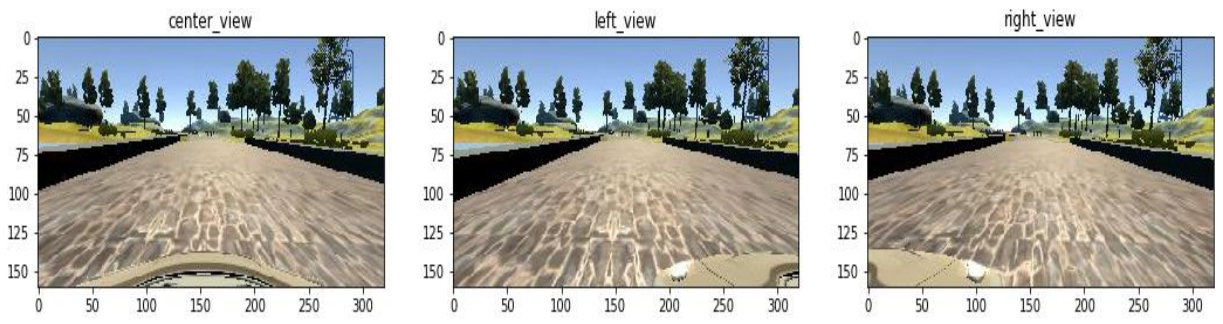
total no of examples is 48216

no of training examples is 38572

no of testing examples is 9644

image input shape (160, 320, 3)

Random examples of the training data:



Design and Test a Model Architecture

Preprocessing:

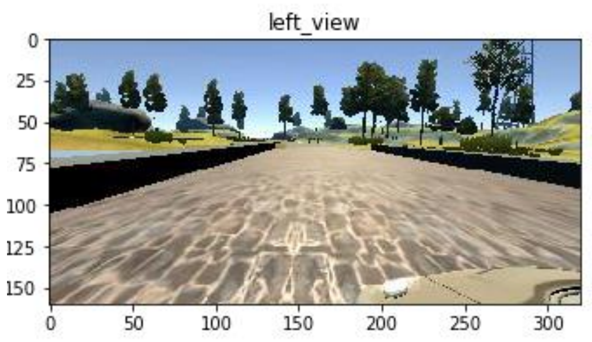
The training data provided by udacity were very good and covered a lot of corner cases

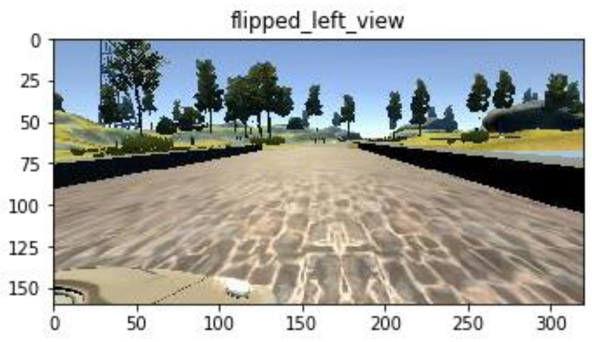
Although I used only the training data provided by udacity and I didn't drive a car,bu I did good preprocessing to this data to make use of the most info I can get using the following steps:

1-Instead of feeding only the center image,I used 3 images from 3 different cameras for the same view

2-I flipped right and left images to increase the training data after inversing the sign of the corresponding steering wheel

Example of this concept:





3-I added correction to the left and right images in order to let the car takes a corrective action when it's not driven in a straight line

4-I normalized the images by dividing pixel values over 255 and decreasing 0.5 using Keras Lambda layer

5-I cropped the top and bottom pixels to give more clear pictures of the road to the car without car bumper nor tree and green area,

I decreased 74 pixels from the top,20 px from the bottom,60 px from right and 60 from right

As we can see we converted the original input shapes of 160,320,3 to 66x200x3 which will be important for feeding the Nividia popular network

6-I flatten the data to prepare it for the Model architecture

Model Architecture:

I used Nividia popular convolutional neural network as shown in the following shape:

1-I used 5 convolutional layers with different no of filter and sizes as shown in the below figure,first 3 layers with VALID padding and the remaining with SAME padding with relu activation function

2-i added dropout layer after each convolutional layer with keep_prob = 0.5 to make sure that I will decrease overfitting

3-I followed the convolutional part with 4 fully connected layers with different shapes as shown in the below figure

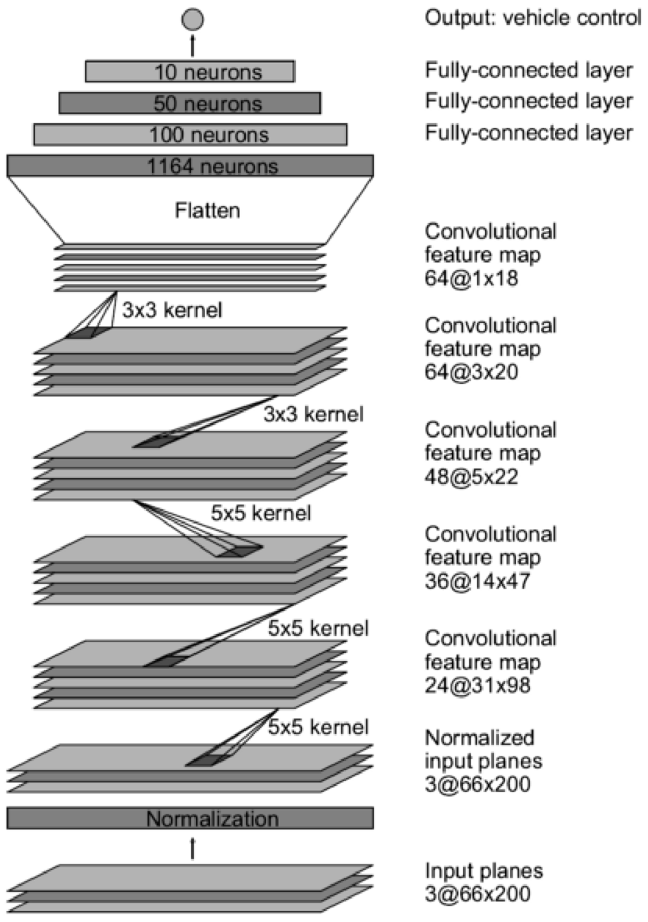
4-i used data_generator() to split the data into batches,I used batch_size = 32

Becaues I I did a lot of images augmentation,I couldn't let the data_generator() read and process image on the fly because the GPU ram was not enough,but I had very good cpu ram(16 gb) which was very enough to do the preprocessing job

5- I used mse(mean square root) as loss function because cross entropy is not relevant for getting steering angle as output

6-i used adam optimizer which is very good optimizer

7-I trained the network with 5 epochs because when I increase it the loss started to increase



Validating the netowrk:

As I mentioned above I splitted the total examples to 80% training data and the remaining for verification

Test a Model on New Images:

I opened the simulator on automatous mode and drive the car with the saved weight after the training and the video is declaring the result

Potential shortcomings with your current pipeline

add more tracks to the training data