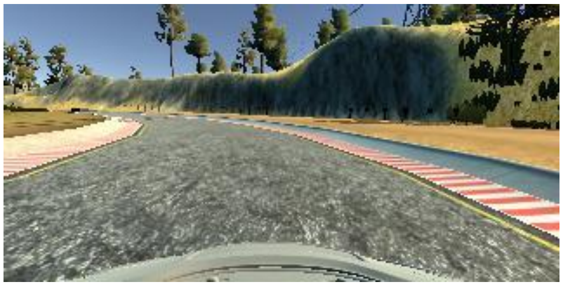
Write Up

In this Write up I will describe how I managed to finish my project behavioral cloning. Firstly, when I downloaded the project, I also downloaded the provided dataset for this project. Then I tried to add my own on it, but there were some problems with Path of images. So, I deleted some of them, but when I tried to load again images, I got not finding the images for corresponding csv names. I added some exceptions, and everything went well. I trained model and was ready to test it on simulator. When I ran it, everything was going good until the first terrain, where there is no yellow line. Unfortunately, my car failed when faced that terrain. Following 2 days I was trying to improve my model, using some opencv functions, another CNN models but nothing worked. Everything was failing on that terrain. So, I went through whole videos and hints on Udacity. And I understood that data is more important for this project than model itself. Then I remembered about deleting some images and some files from CSV, I opened the image folder and I discovered that there were not enough images (almost 0) of that terrain, so how my model could have guessed what to do there. After that I decided to create new dataset on my own, and it worked. Now I will go through all the steps that I've made while developing this project.

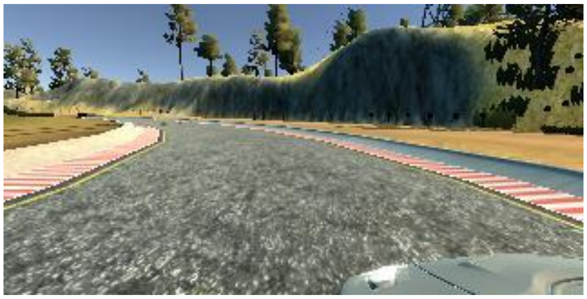
Data:

As I've already mentioned, I've created my own dataset. I drive through the track 4 times. And I gathered around 8500 on camera pictures.

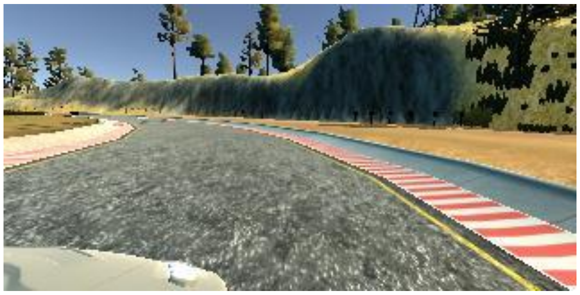
This is the center camera image from our dataset.



Left camera image:



Right camera image:



I use center, left and right camera images. I add 0.2 steering to left camera images and subtract 0.2 steering to right image.

I load center, left and right camera images together. Then to have more data I flip the images using np.flip() function and I multiply steering by (-1) . I add them to my data and finally we have good number of pictures to train our model. But, first as we see there are some features which are not necessary for our model. Like top mountains, and the lower part of image where we can see the car. To get rid of that we first normalize our image, to fit between (-1,1). And then crop the image using Keras Cropping function. **model.add(Cropping2D(cropping=((50,20), (0,0))).** As you see I'm taking 50 pixels from above and 20 pixels from bellow.

Generator

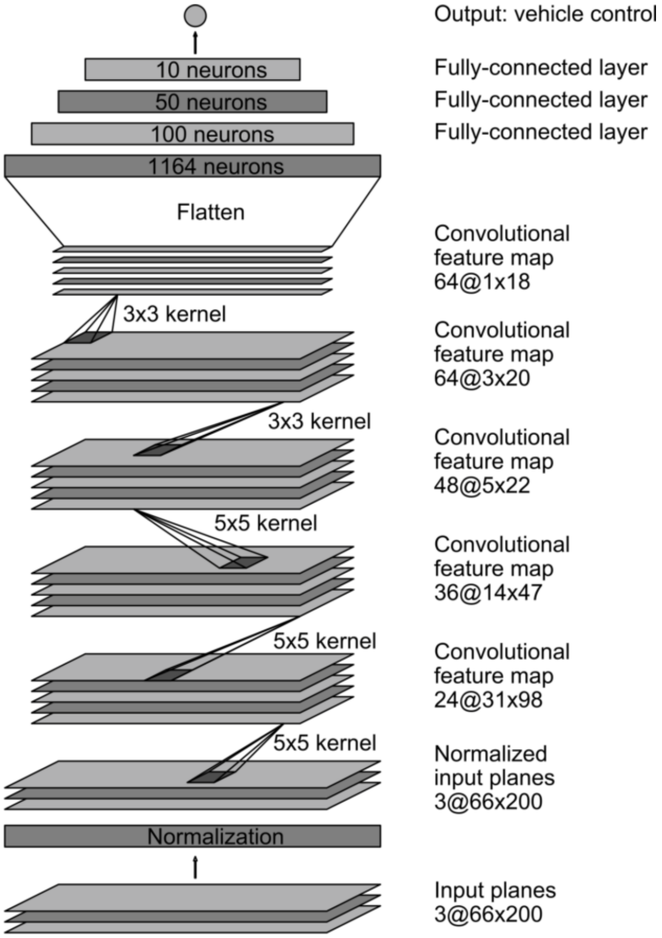
Processing this much amount of data in one flow, is too heavy for computer. Due to that we use Generators to augment our data using batch_sizes and then delete the data that was selected earlier. I used **Batch_size = 32** . Which works well.

Model

As was recommended in Udacity video lectures, I used CNN model released by Nvidia. They define everything about this model in following research paper: **https://arxiv.org/pdf/1604.07316.pdf .**

But I have slightly different parameters. 1) (24, 5, 5) convolution 2) (36, 5, 5) Convolution 3) (48,5,5) convolution 4- 5) (64, 5, 5) convolution. 6) Finally I flatten the neurons 7) Then I have fully connected layer with 100 neurons 8 -9 -10) I Reduce number of fully connected neurons to 50, 10, and 1 for output.

Here is the image of Nvidia CNN model.



To prevent overfitting, I train my NN on 2 epochs because of training time, but 3 or more would be better as I use Dropout() with 50% probability. before flattening my layers and before reducing 100 to 50 layers. which almost guarantees the prevention of the overfitting.

I split my dataset into 80 % training and 20 % validation set. As I've tested it seems most successful approach. **train_samples, validation_samples = train_test_split(samples, test_size = 0.2) .**

The learning rate was not set manually, as I used Adam Optimizer.

Finally, the model was successful at covering the whole 1st trach at maximum speed. Which I changed in drive.py. **set_speed = 30.**

You can have a look at the video provided in my submission.

Note* In the main.py and drive.py I have function preprocess_image(img) which I used earlier for better preprocessing step but it occurred that it was not necessary to make my model successful, so I didn't delete the code, but I don't use it anywhere.**

Further Challenges:

Further challenge will be to cover second track, but I need data for second track as well. After collecting enough data, I am sure my model will be very successful at covering the second track.