Project Writeup/README

Behavioral Cloning

The goals / steps of this project are the following:

- Use the simulator to collect data of good driving behavior
- Build, a convolution neural network in Keras that predicts steering angles from images
- Train and validate the model with a training and validation set
- Test that the model successfully drives around track one without leaving the road
- Summarize the results with a written report

Files/Folders included:

- ✓ `model.py`
- ✓ `model.h5`
- ✓ `video.mp4`
- ✓ `drive.py`
- ✓ `writeup_report.pdf`

RUBRIC POINTS:

1. Required Files

- Are all required files submitted?
  Yes, the submission includes a model.py file, drive.py, model.h5, a writeup report (that you are reading) and video.mp4

2. Quality of Code

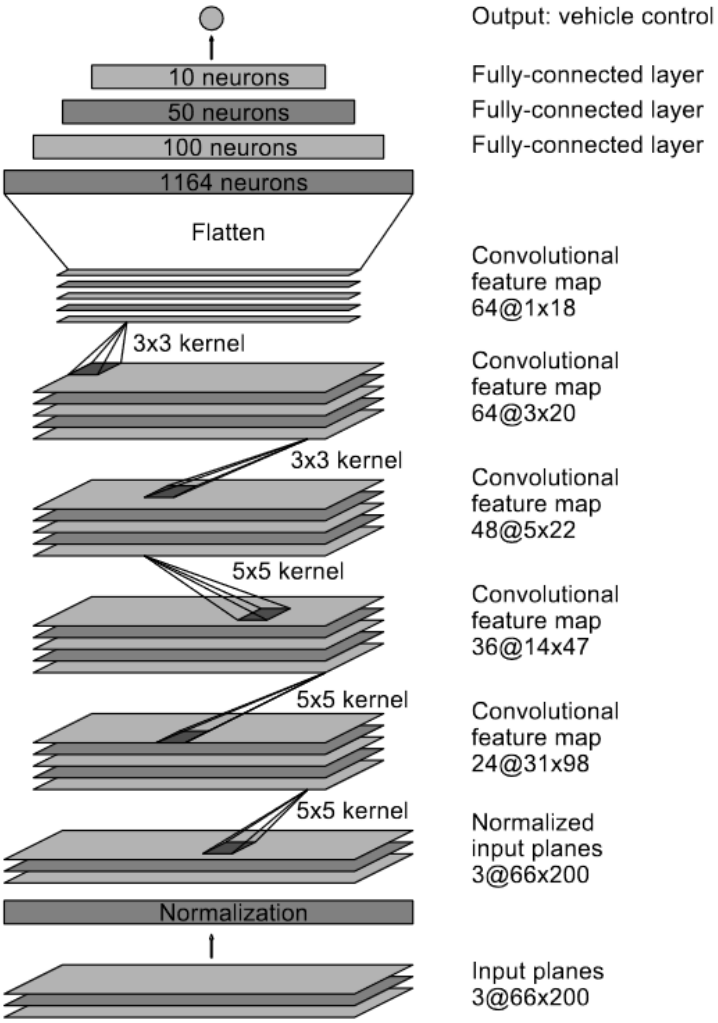
- Is the code functional?
  Using the Udacity provided simulator and my drive.py file, the car can be driven autonomously around the track by executing…

```
python drive.py model.h5
```

- Is the code usable and readable?
  `model.py` can be executed anywhere Python where the environment mentioned in Udacity CarND Term1 Starter Kit is present along with an update in Keras.
  The code in `model.py` uses a Python generator to generate data for training rather than storing the training data in memory. The `model.py` file contains the code for training and saving the convolution neural network. The file shows the pipeline I used for training and validating the model, and it contains comments to explain how the code works.

3. Model Architecture, Data and Training Strategy

- The model architecture has been inspired from the Nvidia's End to End Learning for Self-Driving Cars paper…
  https://images.nvidia.com/content/tegra/automotive/images/2016/solutions/pdf/end-to-end-dl-using-px.pdf

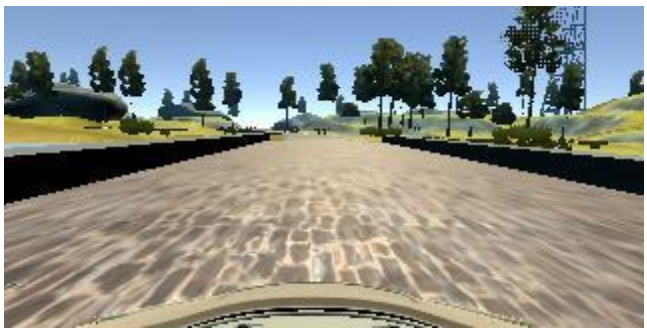


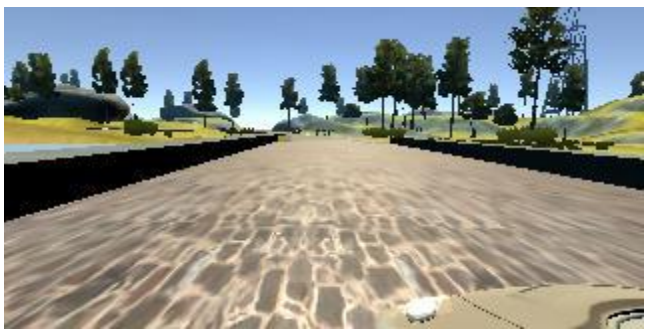
- The model summary is as follows(obtained after training the model)…

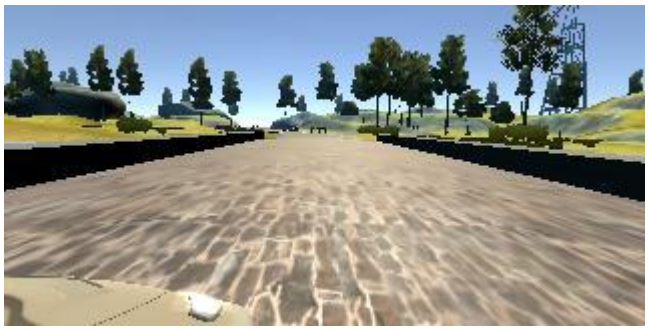
Layer (type)	Output Shape	Param #
lambda_5 (Lambda)	(None, 160, 320, 3)	0
cropping2d_5 (Cropping2D)	(None, 65, 320, 3)	0
conv2d_21 (Conv2D)	(None, 31, 158, 24)	1824
activation_33 (Activation)	(None, 31, 158, 24)	0
conv2d_22 (Conv2D)	(None, 14, 77, 36)	21636
activation_34 (Activation)	(None, 14, 77, 36)	0
conv2d_23 (Conv2D)	(None, 5, 37, 48)	43248
activation_35 (Activation)	(None, 5, 37, 48)	0
conv2d_24 (Conv2D)	(None, 3, 35, 64)	27712
activation_36 (Activation)	(None, 3, 35, 64)	0
conv2d_25 (Conv2D)	(None, 1, 33, 64)	36928
activation_37 (Activation)	(None, 1, 33, 64)	0

```
flatten_5 (Flatten)          (None, 2112)              0
_________________________________________________________________
dense_17 (Dense)             (None, 100)               211300
_________________________________________________________________
activation_38 (Activation)   (None, 100)               0
_________________________________________________________________
dropout_5 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_18 (Dense)             (None, 50)                5050
_________________________________________________________________
activation_39 (Activation)   (None, 50)                0
_________________________________________________________________
dense_19 (Dense)             (None, 10)                510
_________________________________________________________________
activation_40 (Activation)   (None, 10)                0
_________________________________________________________________
dense_20 (Dense)             (None, 1)                 11
=================================================================
Total params: 348,219
Trainable params: 348,219
Non-trainable params: 0
_________________________________________________________________
```

- Dropout layer has been added to the model in order to reduce overfitting
- The Exponential Linear Unit(elu) activation function has been used due to it's promising accurate results. Inspired from the research…
  https://arxiv.org/pdf/1511.07289.pdf
- Adam Optimizer has been used…
  https://arxiv.org/pdf/1412.6980.pdf
- Mean Squared Error loss function has been used
- Epochs has been set to 3
- Batch size is 32
- Validation data is 20% of the total data
- Udacity provided sample data has been used for training, which is fairly robust considering the car sojourns the tracks without going off tracks.
- Sample images from the dataset are…. (from center, left, right camera)







- By adding the cropping layer Cropping2D from Keras, the model will automatically crop the input images when making predictions.
- Images have been cropped 70 pixels from top and 25 pixels from bottom.
- A lambda layer to carry arbitrary functions has been used to normalize the images. The lambda layer will also ensure that the model will normalize input images when making predictions.
- Images have been flipped and opposite steering measurement taken so as to generalize the model, make the car involve in appropriate right turns.
- Flip method from the cv2 library has been used to flip the images.
- Correction is applied to left and right camera images as we want the model to think that all the images are coming in from the same camera, hence adding correction to left cam steering measurement and subtracting from right cam steering measurement.
- The model has been trained on local machine and saved as `model.h5`

4. Simulation

- The car has been driven in autonomous mode using the drive.py file and passing the `model.h5` as argument to it.
- A video has been saved of the output as…. `video.mp4` using the `drive.py` file by passing another argument which is directory of the images to be saved and these images are converted to a video using video.py provided by Udacity.
- The car is successful in covering the tracks and does not leave the drivable portion of the track surface.
