Model Architecture and Training Strategy

1. An appropriate model architecture has been employed

My model consists of a convolution neural network with 5x5 and 3x3 filter sizes and depths between 24 and 64 (model.py lines 60-64)

The model includes RELU layers to introduce nonlinearity (code lines 60-64), and the data is normalized in the model using a Keras lambda layer (code line 58).

2. Attempts to reduce overfitting in the model

The model was trained and validated on different data sets to ensure that the model was not overfitting (code line 14). The model was tested by running it through the simulator and ensuring that the vehicle could stay on the track.

3. Model parameter tuning

The model used an adam optimizer, the learning rate is the default value 0.001, while decay=0.75 so that the learning rate can adapt better with more epochs for training (model.py line 73).

4. Appropriate training data

Training data came from the provided Sample Training Data in the Project Resources

For details about how I created the training data, see the next section.

Model Architecture and Training Strategy

1. Solution Design Approach

Since we want to process images, we shall use convolutional neural networks. The task is to train the car to drive on the paved path from those recorded images. Since this task is not an easy one. The CNN we use shall have some depth. I simply chose the one provided in the course since it is sufficient. The architecture is given below:

2. Final Model Architecture

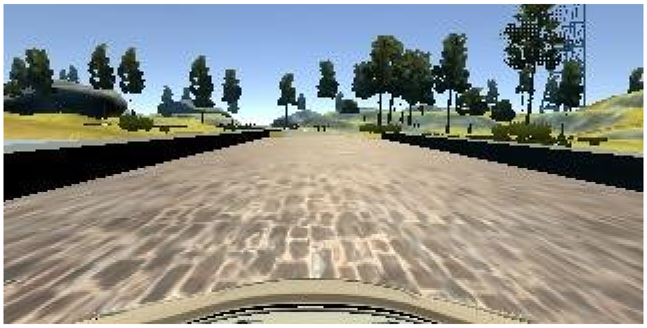
The final model architecture is given below:

```
model = Sequential()

model.add(Lambda(lambda x: x / 255.0 - 0.5, input_shape=(row, col, ch)))

model.add(Cropping2D(cropping=((70, 25), (0, 0))))

model.add(Convolution2D(24, 5, 5, subsample=(2, 2), activation='relu'))

model.add(Convolution2D(36, 5, 5, subsample=(2, 2), activation='relu'))

model.add(Convolution2D(48, 5, 5, subsample=(2, 2), activation='relu'))

model.add(Convolution2D(64, 3, 3, activation='relu'))

model.add(Convolution2D(64, 3, 3, activation='relu'))

model.add(Flatten())

model.add(Dense(100))

model.add(Dense(50))

model.add(Dense(10))

model.add(Dense(1))
```

It is consisted of a convolution neural network with 5 convolutional layers and three essential dense layers. At the beginning, it also provides normalization and cropping for the input images.

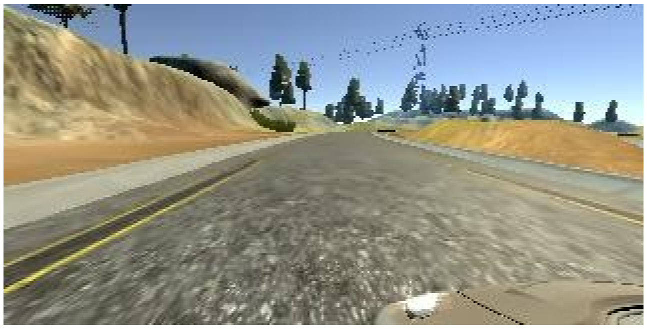
3. Creation of the Training Set & Training Process

To capture good driving behavior, I first recorded two laps on track one using center lane driving. Here is an example image of center lane driving:

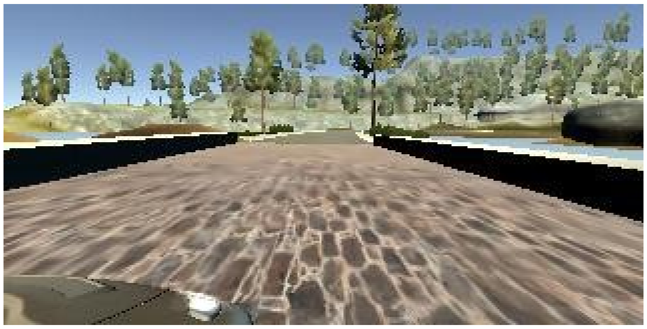


To better aid the car keep on the way and recover from sides, I also used images from the left and right cameras, and adjusted the angle accordingly with a correction of 0.2.

Left image:



right image:



I finally randomly shuffled the data set and put 20% of the data into a validation set.

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 6 or 7 as evidenced by an increase of validation loss. I used an adam optimizer so that manually training the learning rate wasn't necessary.