Problem Statement

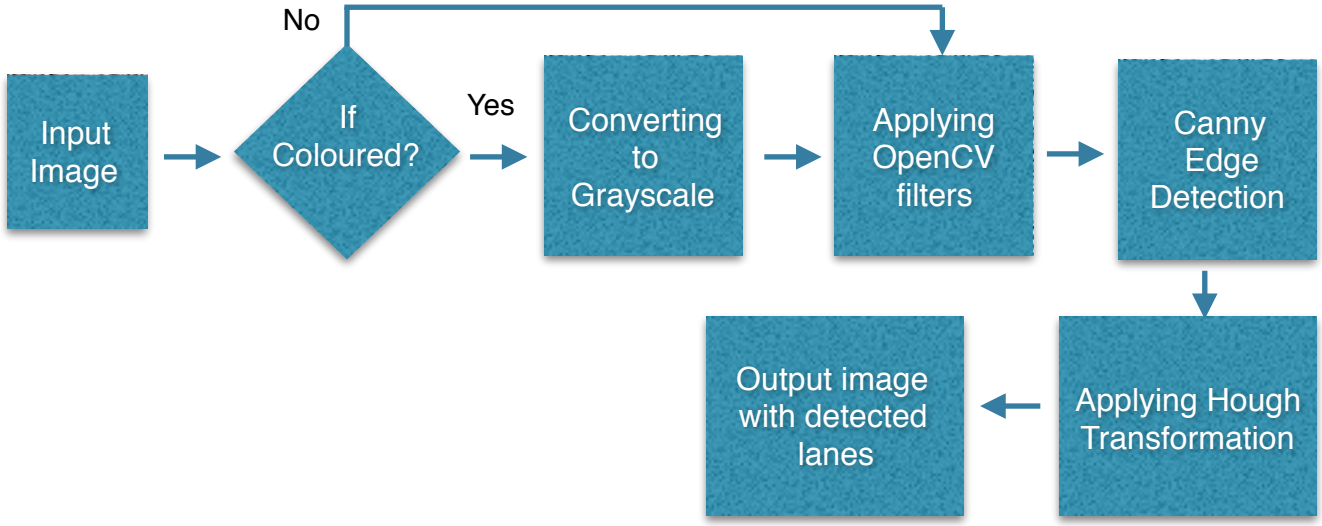
Propose a solution mentioning the approach to automate the lane detection process from street images.

Solution

There are two approaches for lane detection -

Approach-1: Lane Detection using Canny Edge Detection and Hough Transformation (without Deep Learning)

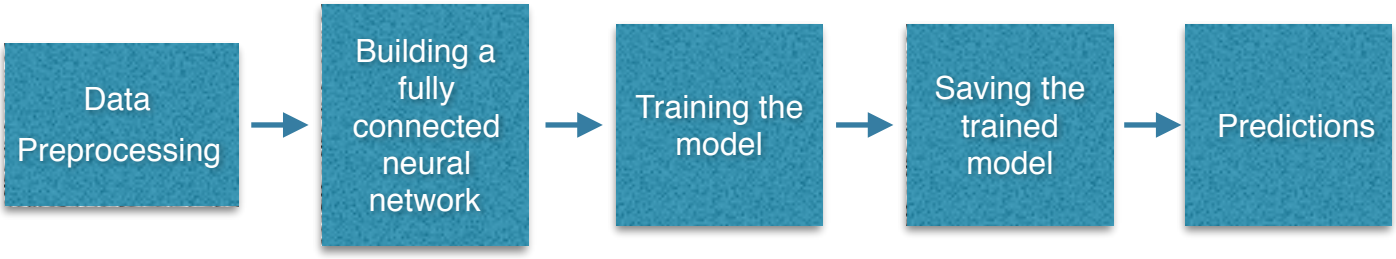
This approach detects lanes by low level feature extractions like lane mark edges. In this approach OpenCV is predominantly used to detect lanes in a street image. An input pipeline consists of techniques applied on the image in a sequence as illustrated below -



Approach-2: Lane Detection using Fully connected CNN (with Deep Learning)

This is the approach that we have followed.

Here, deep learning is used to detect lanes in an image, in order to resolve the problem of identifying curves lane which can't be done by Approach 1. A fully connected convolutional neural network (CNN) built from scratch is used, which is very effective at finding complex patterns within images. This approach is more effective at detecting lanes as well as faster than common computer vision techniques. The approach is illustrated below -



The steps followed are -

Data Pre-Processing : The dataset used for this approach is an open source dataset BDD100K Lane Marking Dataset. This dataset consists of around 500 street images and their corresponding labels.

Building Fully Connected Neural Net : Here, we have built up a FCNN from scratch. The CNN consists of 7 convolutional layers, 3 polling layers, 3 upsampling layer and 7 deconvolution layers, built with Keras on top of TensorFlow. The neural net assumes the inputs in the shape of 80 x 160 x 3 with the labels as 80 x 160 x 1.

Training Model: Before the model training starts the dataset is divided into training set and a validation set created using Sklearn train_test_split method with 90% in the training set and 10% in validation. The model is compiled with mean squared error as the loss function and trained over 50 epochs on the whole training dataset and validated on the validation dataset.

Save Model: After the model training gets completed, this trained model is saved in a json along with its weights. Now this saved model can be used for doing prediction on the unknown data.

Prediction: In this phase the saved model is loaded along with its weights. The test image is pre-processed in the shape of 80 x 160 x 3, as the model expect that as the input shape. The preprocessed image is given as an input to the model which predicts the output. The detected region of lane is marked in Green Colour.

Performances Metrics:

The performances metrics to be used for evaluating the build model are confusion matrix, precision, recall, ROC curve, PR curve, Mean Squared Error (MSE) and mAP score.

Some of the predicted results are as shown below -