**OpenCV (Lane Detection)**

*Submitted in partial fulfillment of the requirements*

*for the award of the degree of*

**Bachelor of Computer Applications**

To

Guru Gobind Singh Indraprastha University, Delhi

Guide: Submitted by:

Dr. Tripti Lamba Akshay Tanwar

Associate Professor Saksham Shrivastava

Anurag Munjal



**Institute of Information Technology & Management,**

**New Delhi – 110058**

**Batch (2017-2020)**

**Certificate**

We, Saksham Shrivastava(03413702017) ,Akshay Tanwar(00113702017) and Anurag Munjal(35113702017) certify that the Summer Training Project Report (BCA-355) entitled “OpenCv (Lane detection)” is done by us and it is an authentic work carried out by us at Institute of Information Technology & Management. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of our knowledge and belief.

Date: Signature of the Students

Certified that the Project Report entitled “OpenCv (Lane detection)”” done by the above students is completed under my guidance.

Date: Signature of the Guide

Countersign Program Director Countersign Director

**ACKNOWLEDGEMENT**

We hereby acknowledge that we have taken efforts in completion of this project. However, it would not have been possible without the kind support and help of many individuals and organizations and we would like to extend our sincere thanks to all of them.

We are highly indebted to **Dr.Tripti Lamba** for her guidance and constant supervision as well as for providing necessary information regarding the project & also for her support during the complete phase of working on the project.

We would like to express my gratitude towards our parents & members of **Institute of Information Technology and Management** for their kind co-operation and encouragement which helped us in completion of this project.

We would like to express our special gratitude and thanks to industry persons for giving us such attention and time.

Our thanks and appreciations also go to our colleagues in developing the project and people who have willingly helped us out with their abilities and knowledge.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S No** | **Topic** | **Page No** |
|  | Certificate | - |
|  | Acknowledgements | - |
|  | List of Tables/Figures/Symbols | - |
|  | Synopsis | I-IV |
|  | Chapter-1: Software Project Planning | 1-10 |
|  | 1.1 Description of the Software System under Study | 1 |
|  | 1.2 Data Collection | 2 |
|  | 1.3 Tools and Platform | 4 |
|  | 1.4 Project Planning | 4 |
|  | 1.5 Methodology | 6 |
|  | * + 1. SDLC Model | 6 |
|  | 1.5.2 Justification for the Selection of Model | 7 |
|  | Chapter-2: Software Requirement Specification | 8 |
|  | 2.1 Description of Information System (Block Diagram) | 8 |
|  | 2.1.1 Product Features | 9 |
|  | 2.1.2 Input/ Output Data elements | 9 |
|  | 2.1.3 Procedures/ rules/ mathematical relationships | 9 |
|  | 2.1.4 Decision Table | 10 |
|  | 2.2 Use Case Diagram | 10 |
|  | 2.3 Software Product constraints | 11 |
|  | Chapter-3: Software Project Analysis | 11 |
|  | 3.1 DFD | 12 |
|  | 3.2 ERD | 12 |
|  | 3.3 Database Specification | 12 |

**SYNOPSIS**

**1. Title of Project**

Lane detection with opencv.

**2. Problems with the Existing System**

Problem with existing system is that they provide good accuracy for high quality images but sometimes provide poor results when are tested under poor environmental conditions like fog , haze , noise , dust etc. For solving this problem we have used canny edge detection which will help us to overcome this problem.

**3. Description of the Proposed System**

**Open CV** is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

**4. Description and identification of the Functional Modules**

In this project we will be going to use different modules

**Def roi(image):** it helps us to select our region of interest.

**Height=image.shape[0]->** here we are defining height at the coordinate where y=0.

**Mask=np.zeros\_like[image]->** here all the pixels values of the image are set to zero.

**Return masked\_image** **->** returns the masked image.

**Def canny(image):** In this first we convert the image into grayscale.

**Average\_slope\_intercept:** in this first we reshape the line then we set the parametres using line coordinates then we find the average intercept and in end this fuction gives the value of coordinates of left and right line.

**Display lines-** In this we set the color of line and its width , it gives highlighted lanes.

**In this project we have used the haarcascade algorithm:: Haarcascade** **algorithm**: it is a machine learning algorithm used to identify objects in an image or video

**5. Tools/Platforms**

**5.1. Hardware specifications**

**At time of training**

RAM 6 GB

HDD 5 GB

Processor I 7

**To Run trained Model**

RAM 4 GB

HDD 1 GB

Processor i3 Intel processor

Webcam

**5.2. Software specifications**

Python 3.7

Open cv3.4.2

Numpy1.1.6

**6. Methodology**

**6.1. SDLC Model to be used**

Prototype model

**6.2. Justification for the Selection of Model**

This model is being used because so with time further improvements can be made in the project.

**7. Future Scope**

In future more objects could be added to the project and the detection would be made more precise.

**CHAPTER 1**

* 1. **Description of the software system under study:**

**OpenCV** is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

IN THIS WE ARE USING A SOFTWARE AND A SOURCE CODE EDITOR MENTIONED BELOW-

**GUI CASCADE TRAINER-**Interactive app for managing the selection and positioning of rectangular ROIs in a list of images, for specifying ground truth for training algorithms, and for creating new cascade classifiers.

**VS CODE-**Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS

* 1. **Data collection-**.

[*http://www.kaggle.com/jessicali9530/stanford-cars-dataset#cars\_test.zip*](http://www.kaggle.com/jessicali9530/stanford-cars-dataset#cars_test.zip)

[*http://image-net.org/api/text/imagenet.synset.geturls?wnid=n06874185*](http://image-net.org/api/text/imagenet.synset.geturls?wnid=n06874185)

**1.3Tools/platforms**

**Hardware specifications**

**At time of training**

RAM 6 GB

HDD 5 GB

Processor I 7

**To Run trained Model**

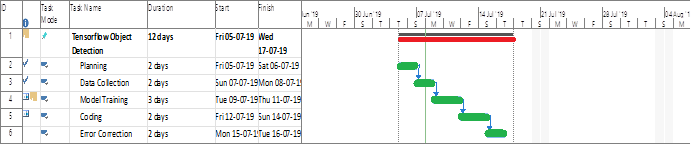
RAM 4 GB

HDD 1 GB

Processor i3 Intel processor

Webcam

**1.4Project Planning**

****

**1.5 Methodology**

**1.5.1. SDLC Model to be used**

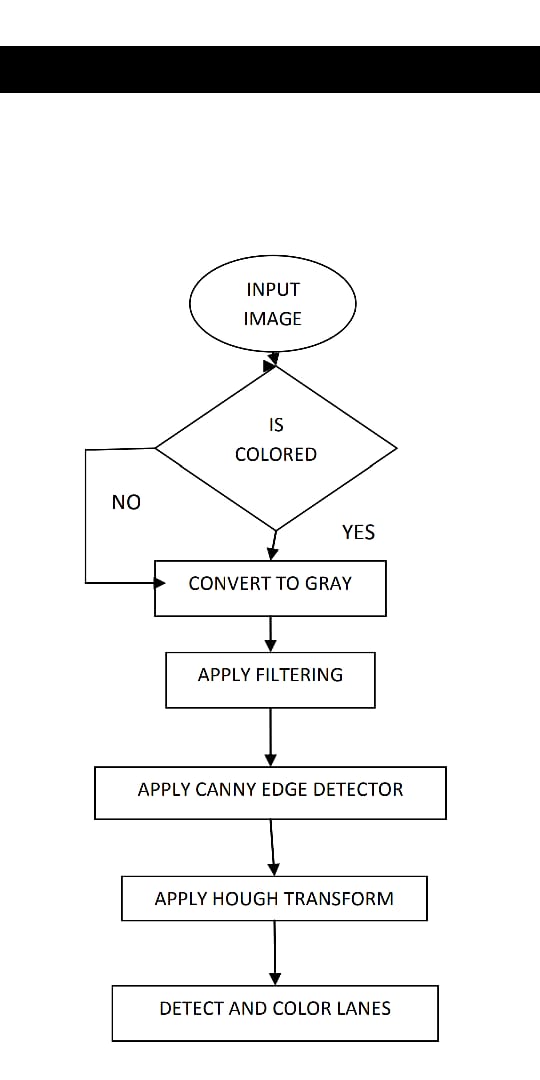
Prototype model

**1.5.2 Justification for the Selection of Model**

So with time we can further improve the model.

**CHAPTER 2**

**2.1Description of Information System (Block Diagram)**: project accepts the user input in the form of voice command that is passed through Natural language processor and classified into a text. On basis of it is a command or an attempt to talk to the voice recognition on the basis of that text is passed to the pre-trained model and then the identified text is being searched through the program and then the desired output according to the text is being shown. the voice is being captured by feature extraction then it is passed to decoder where with the help of acoustic models and language models is being interpreted.



* + 1. ***Product Features*:** An image recognition algorithm takes an image ( or a patch of an image ) as input and outputs what the image contains. In other words, the output is a class label you have to train the algorithm to learn the differences between different classes. If you want to find cats in images, you need to train an image recognition algorithm with thousands of images of cats and thousands of images of backgrounds that do not contain cats. Needless to say, this algorithm can only understand objects / classes it has learned.
    2. **Procedures/ rules/ mathematical relationships***:* A feature extraction algorithm converts an image of fixed size to a feature vector of fixed size. In the case of pedestrian detection, the HOG feature descriptor is calculated for a 64×128 patch of an image and it returns a vector of size 3780. Notice that the original dimension of this image patch was 64 x 128 x 3 = 24,576 which is reduced to 3780 by the HOG descripton

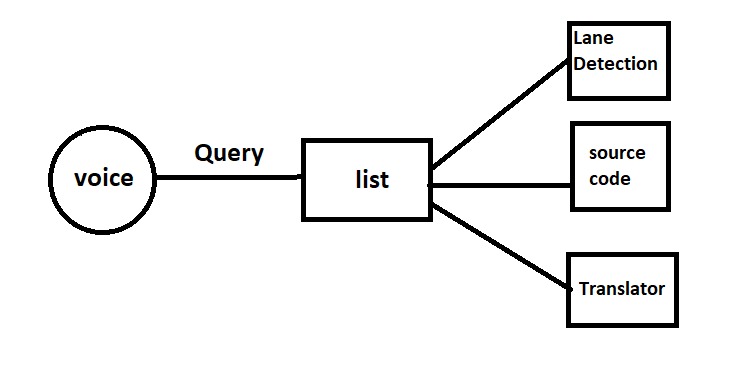
213 **Software Product constraints:** opencv depends on a heavy deep trained model that has to be loaded with the opencv at the start and also uses a proprietary technology of wave net used by google for speech to text which can be accounted as limitations for the opencv where they both act as a dependency for the opencv

**CHAPTER 3**

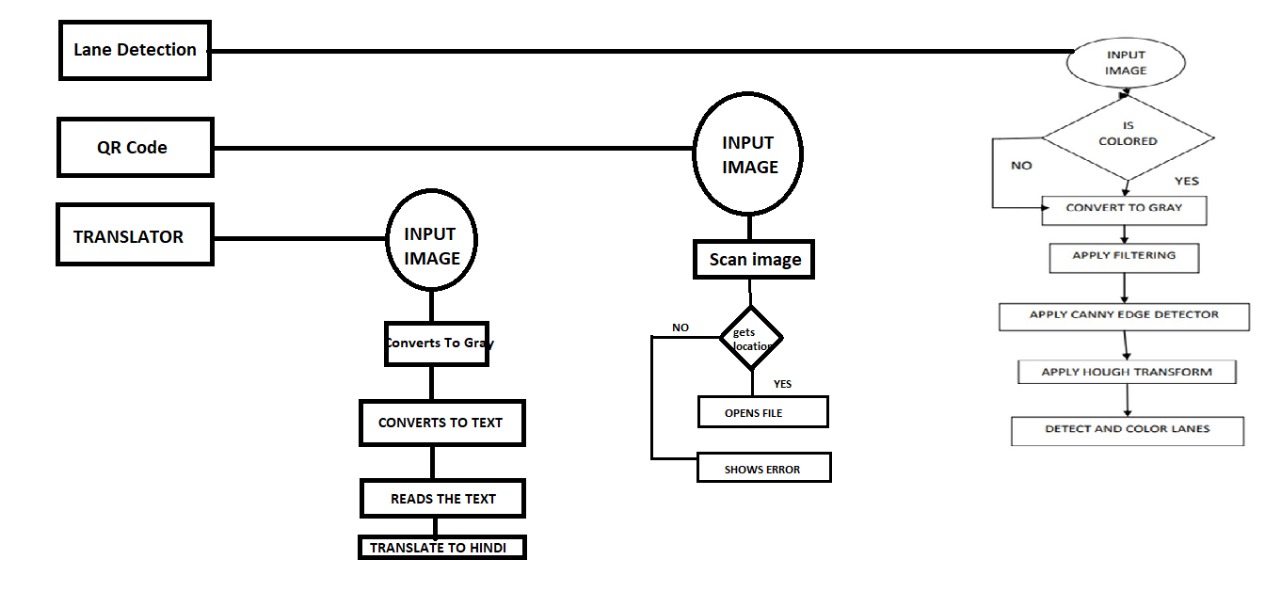
**3.1 DFD (upto 2​nd level) or Class Diagram:**

Voice recognition takes just one input at a time from the user and after processing it just gives the output to the user. But at level 1 it takes the input from the user applies that logic to the pre-trained model and sequence to sequence model. And provides the output to the user.

**LEVEL 0**

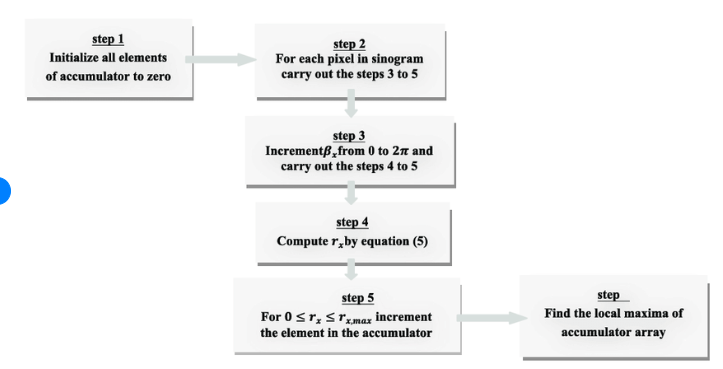


**LEVEL 1**

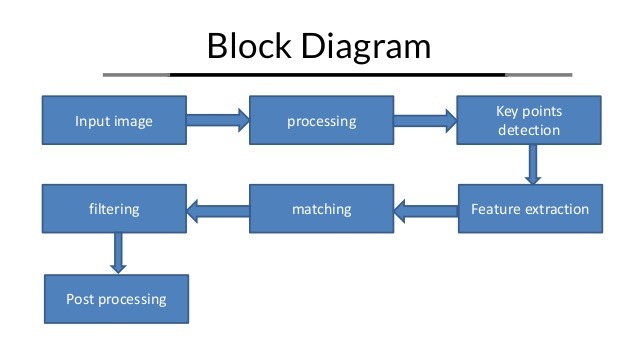


**3.2 Model Specifications**:

Block diagram of algorithm used.



**Block diagram of opencv**

****