Automatic Fare Generation System For Parking Lots

*

Akshay Rai[1] Arayan Kataria[2] Ansh Bhatia[3] Archit Jain[4] Niraj Kumar[5]
Microprocessor And Interfacing , BTech Computer Science Engineering Core
Vellore Institute of Technology , Vellore , Tamil Nadu
[1]akshay.rai2020@vitstudent.ac.in [2]arayan.kataria2020@vitstudent.ac.in
[3]ansh.bhatia2020@vitstudent.ac.in [5]niraj.kumar2020@vitstudent.ac.in [4]archit.jain2020@vitstudent.ac.in

***Abstract*—For parking lots that accommodate a large number of cars daily, our project intends to create an effective and automatic fare generation method. The deployment of human resources is necessary to manage the parking of the visiting cars, estimate the fares, and collect payment at the point of exit in the streets and locations that typically bustle with visitors every day. This project will automate the process, eliminating the need for labour, avoiding deliberate fare overcharging for personal gain, adding an extra layer of security via image and database records, and increasing efficiency by lowering human errors in the process.To put this into practise, the system created must be able to determine the type of vehicle and its identifier, the licence plate. For the same purpose, this project makes use of computer vision and machine learning techniques.**

***Index Terms*—Fare Generation, Machine Learning, Computer Vision, Automobile, Parking System, Deep Learning, Neural Networks**

I. Introduction

The traditional method of collecting fares often entails distributing slips that detail the entry, their collection, and the chosen fare at the time of leaving. This process always takes longer at entry because there are fewer workers and more visitors' cars. Regular users find it to be quite a task to keep the slips that are thereafter given safe and accessible. Furthermore, it frequently happens that employees overcharge customers in order to benefit themselves by generating price slips with false information that violates the established regulations. This poses an additional security risk to the automobiles, as does the lack of any other documentation of the parked cars other than the paper slips.All of these discrepancies not only obstruct the development of a smart parking system but also make the procedure difficult for users. Renovation of the systems to make them as human management independent as feasible is of vital relevance as the world moves toward artificial intelligence.Thus, an autonomous system not only paves the way for hassle-free fare creation but also boosts overall efficiency, saving users' time and money.

II. LITERATURE REVIEW

Computer Vision is the core of Artificial Intelligence which allows computers to take intelligent decisions based on the visual details around them. It includes various takes such as object detection and recognition,image classification and segmentation. Image classification is the task of categorizing images into one of several predefined classes, is a fundamental problem in computer vision. Vehicle classification done using a visual-based dimension estimation method,extracts moving vehicles from traffic image sequences and fits them with a simple deformable vehicle model. A set of coordination mapping functions are derived from a calibrated camera model and relying on a shadow removal method, vehicle's width, length and height are estimated [1].License Plate Detection systems have been employed widely over the world for real time tracking of vehicles, curb criminal activities etc. Various techniques have been developed using different softwares like LabView, MATLAB, OpenCV using different approaches like edge based, neural networks, sliding window techniques amongst others were used. While the initial edge based techniques didn't produce accurate results, with no license plate detection or incorrect detection [2], the techniques developed later on which used median-filters and neural networks, though gave a better accuracy, but could only be used to detect english characters. [3]

III. TECHNOLOGY USED

A. HARDWARE REQUIREMENTS

1) Raspberry Pi 3: Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. It is an SoC (System on chip), which integrates all the components of a computer or other electronic systems on a single chip like a GPU, Wifi-Module etc.Raspberry Pi 3 Specifications are:

- SoC: Broadcom BCM2837
- CPU: 4× ARM Cortex-A53, 1.2GHz
- Broadcom VideoCore IV
- 1GB LPDDR2 (900 MHz)

Fig1: Raspberry Pi 3

2) UltraSonic Sensor (HC-SR04): The human ear can hear sound frequency around 20HZ 20KHZ, and ultrasonic is the sound wave beyond the human ability of 20KHZ.HC-SR04 is an ultrasonic ranging module that provides 2 cm to 400 cm non-contact measurement function. The ranging accuracy can reach to 3mm and effectual angle is ¡ 15°. It can be powered from a 5V power supply. [4]

- Working Voltage : DC 5V
- Working Frequency : 40Hz

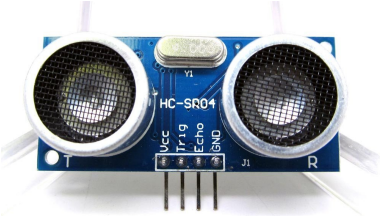


Fig2: Ultrasonic Sensor

3) DC MOTOR: A DC motor in simple words is a device that converts electrical energy (direct current system) into mechanical energy. The direction of current is given by the Fleming's Left Hand Rule.

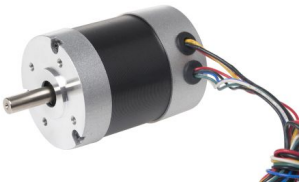


Fig3: DC Motor

B. SOFTWARE REQUIREMENTS

1) TENSORFLOW: TensorFlow is an open source software library for high performance numerical computation, originally developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with strong support for machine learning and deep learning. Owing to its flexible architecture,it allows deployment across different platforms ranging from desktops to mobile and EDGE devices. [5]

2) KERAS: Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano, two of the top numerical platforms in that provide the basis for Deep Learning research and development.

3) Python 3.6.4: Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability.Important library of python used here is OpenCV which is a leading open source library for computer vision, image processing and machine learning, and now features GPU acceleration for real-time operation.

4) Anaconda: Anaconda is a free and open source distribution of the Python and R programming languages for data science and machine learning related applications, that aims to simplify package management and deployment.The packages used from conda distribution as part of this project are :

- Scikit-Learn
- NumPy
- Pandas
- Matplotlib
- Augmentor
- Theano

IV. METHODOLOGY

A. OBJECT DETECTION

When a vehicle approaches the entry barrier, its presence is detected using an ultrasonic sensor. The ultrasonic sensor intimates the processor of the presence of the vehicle which then turns on the camera.The camera captures the image of the vehicle and that image is then used to recognize license plate detection and vehicle classification. Ultrasonic Sensor is the most commonly used sensor to detect a moving target and estimate the the approximate distances it is present at.Ultrasonic sensors works on the principle of emitting short, high-frequency sound pulses at regular intervals. These propagate in the air at the velocity of sound. If they strike an object, then they are reflected back as echo signals to the sensor, which itself computes the distance to the target based on the time-span between emitting the signal and receiving the echo.

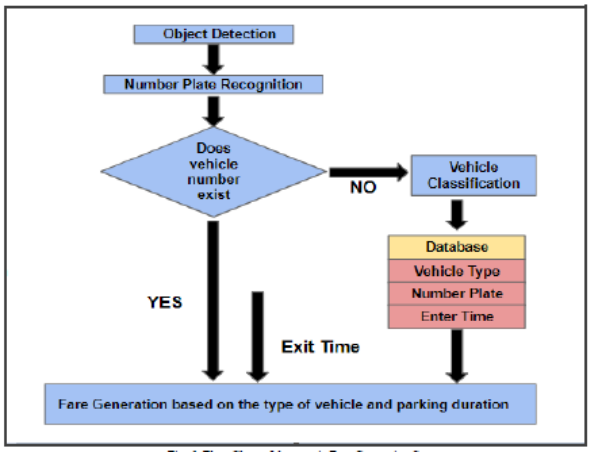


Fig4: Flow Chart of Automatic Fare Generation System

As the distance to an object is determined by measuring the time of flight and not by the intensity of the sound, ultrasonic

sensors are excellent at suppressing background interference. Virtually all materials which reflect sound can be detected, regardless of their colour. Even transparent materials or thin foils represent no problem for an ultrasonic sensor. [6]

B. LICENSE PLATE RECOGNITION

License Plate Recognition uses the concepts of Digital image Processing and Optical Character Recognition to extract the registered vehicle number from the license plate. This procedure is divided into three steps:

- Plate Detection
- Character Detection in Plates

1) Plate Detection: The image is first converted into its grayscale equivalent and then into binary image. A certain threshold is decided and all gray level values above the threshold are mapped to 1 and below it to 0.Thresholding an image converts it into one with black background with all high frequency components in white.Y=0 ;GL less than Threshold and 1 ; GL greater than equal to Threshold

Fig5: Cropped number plate from original car image

Contours are then drawn for all the objects in the binary image using openCV commands findContours() and drawContours(). A contour is a numpy array of coordinates(x,y) of all the points forming the boundary of images. findContours() returns a list of all the contours in the image;the retrieval mode and contour approximation method are set by the programmer, in this case the same being simple chain approximation(cv2.CHAINAPPROXSIMPLE).The list created using findContours() is give as argument to drawContours() which draws contours around all objects of the list. Possible characters are detected from the above image by comparing the dimensions,area and aspect ratio of the bounding rectangle of possible character to predefined pixel dimensions, area and aspect ratio.

```
if (possibleChar.intBoundingRectArea > MIN_PIXEL_AREA and  possibleChar.intBoundingRectWidth > MIN_PIXEL_WIDTH and
possibleChar.intBoundingRectHeight > MIN_PIXEL_HEIGHT and   MIN_ASPECT_RATIO < possibleChar.fltAspectRatio and
possibleChar.fltAspectRatio < MAX_ASPECT_RATIO):

    return True

else:

    return False
```

Fig6 : Code snippet

If the above condition are satisfied it adds the characters to a list called listofPossibleChars. Each possible character is compared with the listofPossibleCharacter to find matching Characters. Distance and angle between characters, change in area, height and width is calculated to do the same.Based on the length on the listofMatchingCharacters found, a new list, ListofListsofMatchingCharacters, of all matching character sequences which could be possible plates is created.From this list, possible plates are extracted and cropped.

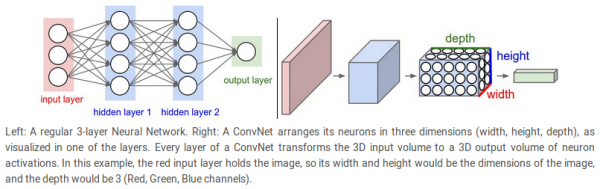
2) Character Detection in Plates: All possible plates are resized for the purpose of clarity and preprocessed. The image is inverted, converted to gray scale and then into binary by thresholding. Possible characters are detected as in the case of plate detection following the exact same steps.After generating the ListofListsofMatchingCharactersinPlate, the overlapping characters are removed. The longest length of matching chars is chosen to be the actual licence plate. The characters are recognized using KNN (K Nearest Neighbor) classification model.

Fig7 : Number plate extraction

3) VEHICLE CLASSIFICATION: Vehicle classification is performed using Convolutional Neural Networks(CNN or ConvNets) which are a special kind of multi-layer neural networks. Convolutional Neural Networks are designed to recognize visual patterns directly from pixel images with minimal preprocessing.We have used pretrained Residual Network deep learning architecture and then fine-tuned the network to fit our dataset of images of Cars and Bikes.

- Convolutional Neural Network : Convolutional Neural

Networks take advantage of the fact that the input consists of images and they constrain the architecture accordingly. [7] Unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. Moreover, the neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner [7] Basically, a ConvNet is made up of Layers. Every Layer has a simple API: It transforms an input 3D volume to an output 3D volume with some differentiable function that may or may not have parameters [7]



Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

Fig8 : Neural Network and Convolutional Neural Network

- Residual Network : Residual Network, popular as ResNet is the most groundbreaking work done in the Computer Vision/Deep Learning community in the last few years [8].Developed by Microsoft, ResNet, is a residual network framework which makes training of deep neural networks easier by eliminating the problem of vanishing gradient and performance degradation.
- Database Management : Csv (Comma Separated Values) file is used to store Number Plate of the vehicle ,Type of Vehicle (Car or Bike) ,Entry Time of the vehicle.This is implemented using Pandas in Python.

V. IMPLEMENTATION

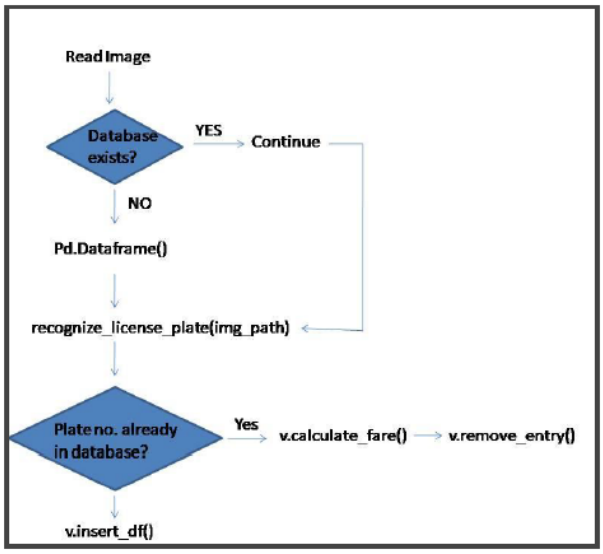


Fig9 : Code Flow

As soon as ultrasonic sensor detects the object,camera clicks the image and that image is passed as argument to the 'main' function . The main function opens the database of the vehicles or creates one if it doesn't exist already.The license plate is recognized and stored in variable licenseplate .An object of class FareCalculator is created. Image, dataframe and license plate are the members of class which get instantiated upon making of the object.The record of license plate just recognized is checked in the database via the method 'checkexistence()'.

- If record exists, Fare is calculated using 'calculatefare()' and printed on the screen.The record is removed via 'removeentry()' method.
- Else if record does not exist, Vehicle type is identified using the method 'testtype()'.The vehicle information like license plate number,vehicle type and entry time are inserted in the database by 'insertdf()' method.

VI. RESULTS

Vehicle classification trained with 672 photos and obtained training accuracy of 94% and validation accuracy of about 92%. By adjusting the hyperparameters more precisely and lowering the regularisation, this accuracy can be raised even more.

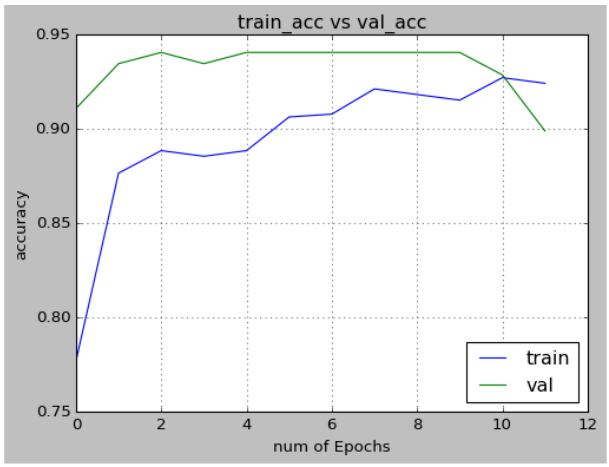


Fig10 : Training v/s Validation accuracy for vehicle classification

The majority of the time, License Plate Recognition can recognise number plates on cars, but it can be difficult to reliably identify plates on bikes. Instead of utilising K Nearest Neighbor, which has a high time and space complexity and produces results that are somewhat erroneous, SVM (Support Vector Machine) classifier can produce results that are more accurate.

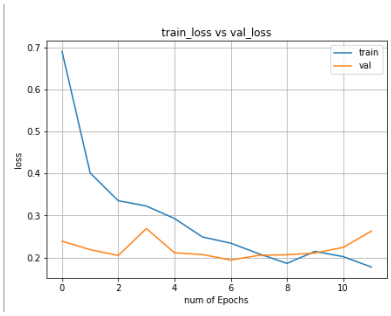


Fig11 : Training v/s Validation loss for vehicle classification

The whole process takes around 25 seconds on an Intel Core i5-4210U 1.70GHz CPU which can be further optimised by using Graphical Processing Unit.

VII. CONCLUSION

An automatic fare generation system can be used to build a parking system that is effective and quick. However, the system can be enhanced by using a licence plate detection system with an SVM model rather than a KNN to get better accuracy.The project can be extended to develop an app to convert traditional parking spaces into a smart parking lot. The app functionality will include the following

Each user can have an account with single or multiple license plates registered. The user will get all information regarding payment and receipts available in his account. Machine learning can further be deployed to give discount to the users on the basis of their usage of that particular parking lot.The functionality of the app will consist of the following : Each user is allowed to register one or more licence plates on their account. The user will receive all payment and receipt-related information that is available in his account. Machine learning can also be used to provide discounts to customers based on how often they utilise a certain parking lot. Fare can be automatically deducted by syncing account information and linking Aadhar.

REFERENCES

[1] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[2] A. Lai, G. Fung, and N. Yung, "Vehicle type classification from visual-based dimension estimation," in *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No.01TH8585)*, pp. 201–206, 2001.

[3] H. Jung, M.-K. Choi, J. Jung, J.-H. Lee, S. Kwon, and W. Young Jung, "Resnet-based vehicle classification and localization in traffic surveillance systems," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 61–67, 2017.

[4] Y. LeCun *et al.*, "Lenet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet*, vol. 20, no. 5, p. 14, 2015.

[5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[6] J.-W. Hsieh, L.-C. Chen, D.-Y. Chen, and S.-C. Cheng, "Vehicle make and model recognition using symmetrical surf," in *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pp. 472–477, IEEE, 2013.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition. cvpr. 2016," *arXiv preprint arXiv:1512.03385*, 2016.

[8] A. Karpathy, "Stanford university cs231n: Convolutional neural networks for visual recognition," *URL: http://cs231n. stanford. edu/syllabus. html*, pp. 13–35, 2018.