Vehicle Identification in Videos

Tanvi Vidhate, 820851740, Meng Zhou, 813349193, Raghav Kishan Sunku Ravindranath, 820174908, Niranjan Hegde, 820625579, Shweta Kisan Shahane, 820899242

***Abstract*—With the increase in number of vehicles on campus, there arises a need to provide an automated system to provide students and staff with the information about the availability of parking space in a parking lot, thereby reducing the time taken to find empty parking spaces with minimal human effort. In this project, we will be exploring options of solving the above stated problem using computer vision and machine learning techniques. We are using a video camera to record the vehicle movement in a parking lot. These videos are subjected to data organization techniques followed by computer vision and machine learning methods to detect and count the number of vehicles that enter and exit the parking lot. To decrease the cost of implementation for existing parking lot structures, an integrated vision-based system is an excellent choice.**

***Index Terms*—Histogram of Oriented Gradients (HOG), K – Nearest Neighbors (KNN), Scale - Invariant Feature Transform (SIFT), Support Vector Machine (SVM)**

I. Introduction

FINDING empty parking spaces in a parking lot can be an cumbersome task. A system that can automatically identify empty parking spaces and guide users to it which will save a lot of time, money and effort. There are many solutions to resolve this issue, some which would employ integrated sensors, some which would use different colored lights for each parking space, but they are either too expensive to implement or have failed to be efficient. This encouraged us to come up with a computer vision based technique, which would use cameras placed at the entry and exit points of the parking lot. These cameras will be used to record a video footage of all vehicles entering and exiting the parking lot. We will then process this footage to count the number of vehicles entering and exiting the parking lot. This approach provides a cost-effective solution to this problem as it doesn't require any major infrastructure change, we can use existing infrastructure to deploy security cameras and interconnect them via a network and save to a central server. A detection and counting application can be created using a video camera and a normal computer.

To process the video footage, we use multiple image processing methodologies such as blob detection and analysis, background-foreground subtraction coupled with supervised machine learning techniques using feature vectors to detect and predict the count of vehicles in a video. These feature vectors are generated using different techniques to obtain maximum accuracy.

II. Task Description

The process of object detection and counting is performed by following the below mentioned steps.

1) Obtaining the video footage for ground truth – cameras were fit at the entry and exits points of a parking lot to record the movement of cars from and to the parking lot.

2) The recorded videos were then annotated using VATIC (Video Annotation Tool). These annotated videos are used to generate the ground truth files which are being used for training, validating and testing machine learning models.

3) The ground truth data consists of car Id, Xmin, Ymin, Xmax, Ymax, frame number, lost, and occluded. This information is useful to implement the machine learning models.

4) Multiple machine learning algorithms are applied on the training & testing data.

5) These models are validated using the validation data set and tested using the testing data set.

6) The machine learning techniques used in this project for vehicle identification are Blob Detection & Analysis, Linear SVM, RBF SVM, KNN.

7) The above-mentioned algorithms are supplied with different feature vectors such as black & white ratio, HOG, SIFT.

8) The black & white ratio feature vector is generated by diving a frame of a background subtracted video into blocks & cells. Then the ratio of count of black to white pixels are taken to form the feature vector.

9) The Histogram of Oriented gradient feature vector is generated using the HOG () function of MATLAB.

III. Major Challenges

1) To obtain permissions from the parking lot committee to setup cameras.

2) To gather the ground truth video footage, in different weather and light conditions.

3) Annotating the many videos to obtain the ground truth data.

IV. Experiment

Vehicle Detection Using BLOB [7]

A. Dataset Description

The data set being used here is obtained from cameras fixed at the entry and exit points of parking lots. These cameras record videos of resolution 720 * 1024 pixels with 60 frames per second. These videos are resized to increase the processing speed of algorithm. [1]

B. Evaluation Metrics

To visualize the performance of this algorithm, the total count of identified vehicles for the test videos is compared with the ground truth data obtained from the annotations.

C. Major Results

As described in the Evaluation Metrics section the counts of cars obtained from the Blob analysis results which is then compared with the counts obtained from the ground truth. It is observed that the accuracy is about (90%) which is fairly accurate.

D. Analysis

Blob Detection techniques are aimed at identifying regions in an image based on certain properties such as such as brightness and color changes in the image. In general, a blob is a region in which some properties are constant.

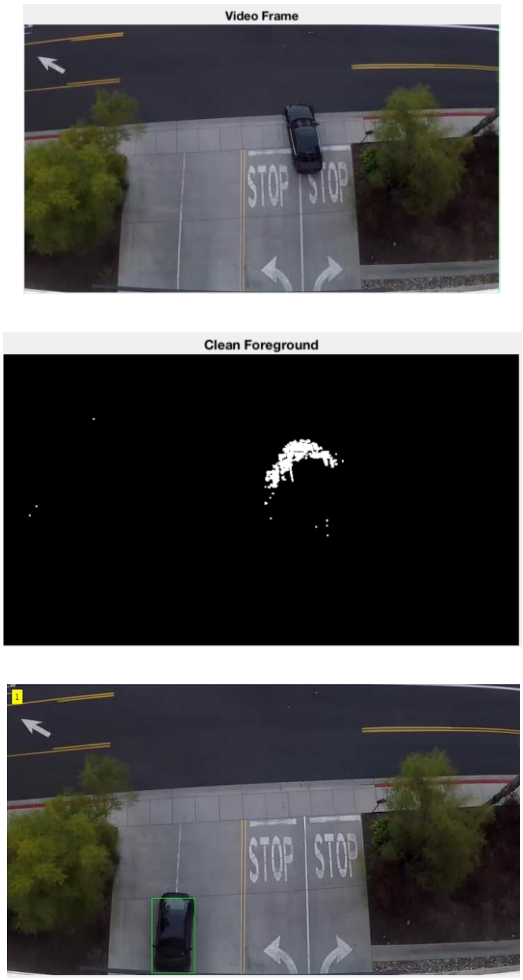
The purpose of BLOB extraction is to isolate the blobs (objects) in a binary image. A blob consists of a group of connected pixels. Whether or not two pixels are connected is defined by the connectivity, that is, which pixels are neighbors and which are not.

An advantage of this method is that it includes high flexibility and good performance. The disadvantage is it requires a clear background-foreground subtraction.

A foreground detector is created using the matlab function called vision.ForegroundDetector () which is supplied with model name – NumGaussians and the number of initial frames of the video to be utilized.

The foreground detector created here is then subjected to the function strel () to reduce the noise in the generated foreground. This filtered foreground detector is then applied on every frame of the video to obtain the foreground of the frame. The foreground which is obtained is then subjected to a BLOB analysis function created which is shown below.

```
blobAnalysis = Vision.BlobAnalysis
('BoundingBoxOutputPort', true, ...
'AreaOutputPort', false, 'CentroidOutputPort', false,
... 'MinimumBlobArea', 2500, 'MaximumBlobArea',
100000);
```
[6]



A car being identified and counted using a foreground detector and Blob Analysis.

The Blob detection technique has a disadvantage. It considers noise within the vicinity of the vehicle. Hence, we are moving forward with other methodologies.

Vehicle Detection Using Black & White Ratio Part A [8]

A. Data Description

We use the video taken from the parking structure and split them up into number of frames. These frames are used for training. Parking structure video is used for testing. Ground truth is computed using these videos and accuracy is determined by comparing the predicted results with the ground truth.

B. Evaluation Metrics

Based on the black and white ratio computed we determine the results by comparing with the testing model, then accuracy and confusion matrix is generated

C. Major results

The approach aims in identifying the cars based on foreground subtraction modeling. We use two different labels namely one car, two car and no cars to train our model. We train the model to compute the average, maximum and minimum black to white ratio for each of the labels. We later use this to predict the cars in each frame of the video. The predication is done using the foreground subtraction. With the help of Gaussian mixture model, we compare the background model to a video frame and determine if the individual pixels are part of foreground or background. We later filter all the foreground subtracted images to make all the frames uniform. The number of black to white pixels are computed for the filtered image and then compared with the testing mean observed earlier to predict if a car is present in the frame. The Foreground detector has the following parameters:

1. Adapt Learning Rate: When you set this property to true, the object sets the Learning Rate property to 1/(current frame number). When you set this property to false, the learning rate property must be set at each time step

2. Number of training frames: This specifies Number of initial video frames for training

background model. We have set this to a default value of 150

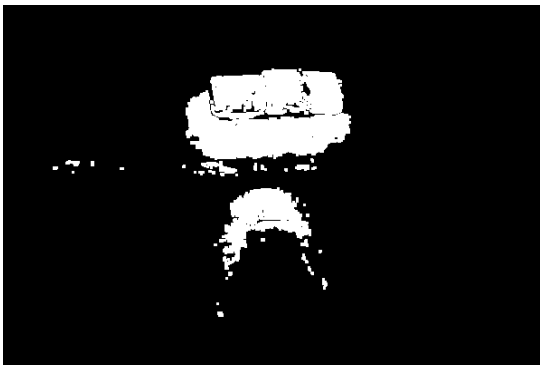
3. Learning rate: This controls how quickly the model adapts to changing conditions. The default ratio is 0.005

4. Minimum black to white ratio: This represent the minimum of the probabilities for pixels to be considered background values.

5. Gaussian Modes: This determines Number of Gaussian modes in the mixture model.

D. Analysis

The model gives a pretty good accuracy when we have to logistically compute to check if the car is present or not in the frame. The average accuracy was 85%. It predicted most of the frames right, however it failed to predict few occluded cars in the frame. Though the model worked really well to classify the frame to detect if it has a car or not, it had an average performance of accuracy of about 65% when we had to detect the number of cars in each frame. This is mainly because of the distortion in the black and white ratio when multiple cars are occluded from the frame.



Vehicle Detection Using SVM & KNN

A. Dataset Description.

Using cameras at fixed at the entrance and exit points of a parking lot, videos of vehicles entering and exiting the parking lot have been recorded. Few videos from these recording have been annotated to generate the ground truth. [1]

The above data set is separated into training, validation and testing data set. The videos are recoded with the resolution of 720 * 1024 pixels with 59 frames per second.

B. Evaluation Metrics

To visualize the performance of an algorithm, typically a supervised learning a confusion matrix is used. Also, known as error matrix, each column of the confusion matrix signifies an instance of a predicted class and each row signifies an instance of the actual class.

		Prediction	
		$\hat{y}=1$	$\hat{y}=0$
Groundtruth	$y=1$	True-positive	False- Negative
	$y=0$	False-positive	True-negative

The above table is an example of a confusion matrix. If the prediction and ground truth are equal, then it is either True-positive or True negative based on the classification labels. If the prediction is not equal to the ground truth, then it is either False-positive or False-Negative based on the classification labels.

The value obtained for the confusion matrix for our algorithm is as follows:

C. Major Results

Using Linear and RBF SVM:

Scale Invariant Feature Transform – SIFT:

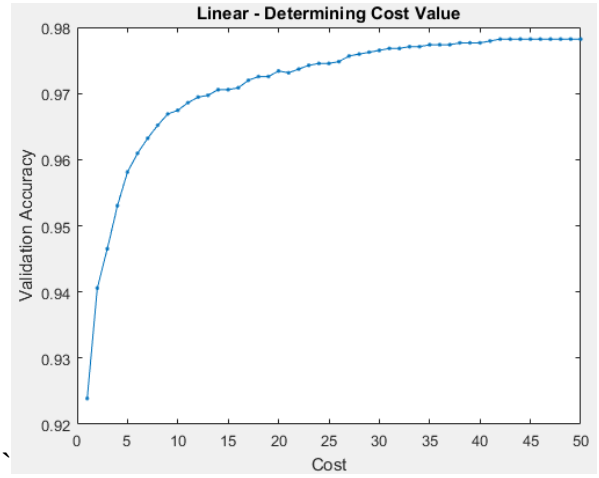
Validation data set
Linear SVM: Accuracy: 97.82%
RBF SVM: Accuracy: 98.56%

Testing data set:

Linear SVM:

Predictions					
Ground Truth		0	1	2	3
	0	2127	6	0	0
	1	47	1159	2	0
	2	2	26	113	0
	3	0	0	0	51

Accuracy = 97.65%

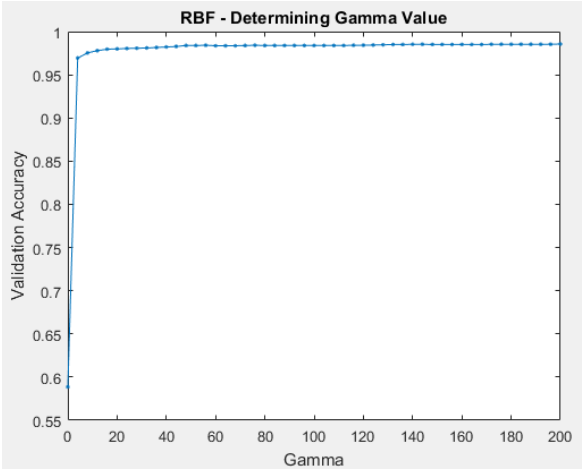


Best cost = 42

RBF SVM:

Predictions					
Ground Truth		0	1	2	3
	0	2130	3	0	0
	1	35	1171	2	0
	2	1	7	133	0
	3	0	0	0	51

Accuracy = 98.64%



Best gamma = 200

Black and White Ratio:

Validation data set:

Linear SVM: Accuracy: 89.02%
RBF SVM: Accuracy: 95.39%

Testing data set:

Linear SVM:

Predictions					
Ground Truth		0	1	2	3
	0	2150	17	0	0
	1	315	879	7	0
	2	15	28	87	0
	3	0	0	1	34

Accuracy: 89.16% (Best cost = 200)

RBF SVM:

Predictions					
Ground Truth		0	1	2	3
	0	2121	13	0	0
	1	118	1104	1	0
	2	3	26	94	0
	3	0	2	4	47

Accuracy: 95.27%

Using K – Nearest Neighbors [3]:

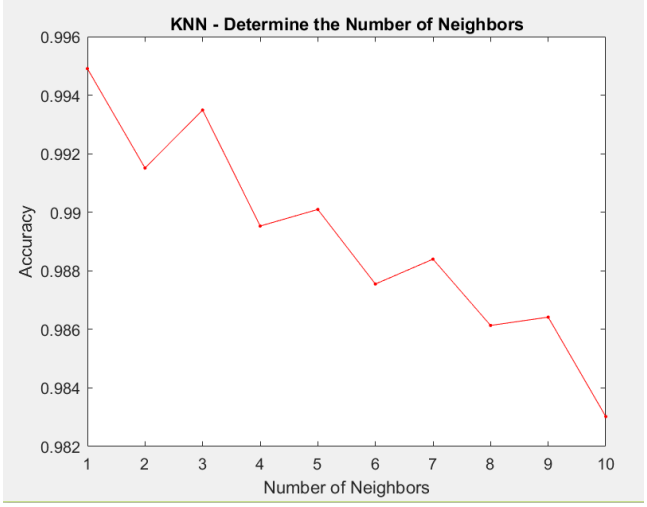
Scale Invariant Feature Transform – SIFT:
Validation data set:

Accuracy: 99.35%

Testing data set:

Predictions					
Ground Truth		0	1	2	3
	0	2106	6	0	0
	1	11	1217	3	0
	2	0	1	147	1
	3	0	0	0	41

Accuracy = 99.37% (K=3)



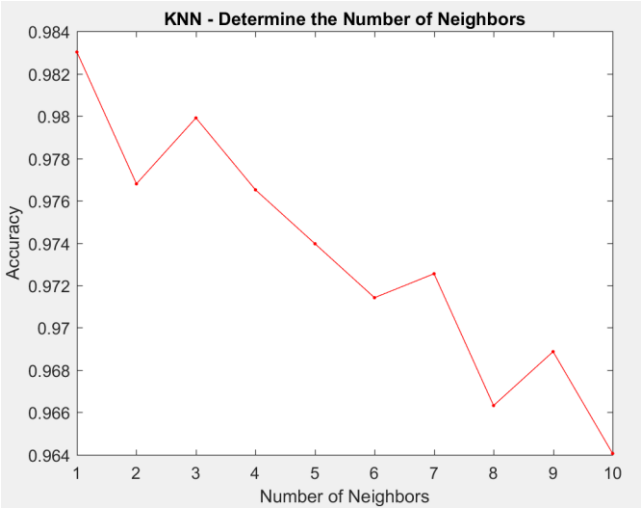
Black and White Ratio:
Validation data set:

Accuracy: 97.990945%

Training data set:

Predictions					
Ground Truth		0	1	2	3
	0	2117	15	0	0
	1	26	1177	7	0
	2	0	7	127	0
	3	0	0	2	55

Accuracy: 98.386640%



D. Analysis

Using Linear and RBF SVM:

The SIFT and black & white feature extraction algorithms are applied on each frame to generate a feature vectors.

SIFT feature Extraction:

1. Load the video and the ground truth annotations (mat) file.

2. Create a video data matrix of size num_frames x (hog_length+1). The last column is reserved for class labels (0, 1, 2, 3 cars and so forth).

3. Extract SIFT features for each frame with a cell size of 128x128 and a block size of 4x4. The sizes are ambiguous. The reason for the cell size and block size to be the specific values above is that it would not generate a result that is either too large or too small.

4. Iterate over the annotations matrix, looking for the corresponding frame numbers both in its cells and in the video data matrix. For every same frame encountered, one is added to the class label.

Black & White ratio feature vector:

1. Frames are extracted from the video using the matlab function videoReader ().
2. Apply background subtraction on the frame.
3. Resize the frame to 256 X 256
4. Divide the frame into a grid of 16 X 16 cellsize.
5. Compute the black and white pixels in each cell.
6. Calculate the black to white pixel ratio for each cell is obtained.
7. Each cell represents a bin in the feature vector.

After obtaining the SIFT as well as black and white ratio feature vectors, the video data from the video and the class labels from the annotations matrix, the data is trained with an SVM and K-NN classifier. LIBSVM [2] 3.2.2 for Linear and RBF SVM, is used for the implementation of both training and testing.

To do the training and testing, the video data was randomly split into a training dataset and a testing dataset with a ratio of 80%-to-20%.

From the above results, it can be inferred that K-NN with SIFT feature set has the highest accuracy of 99.37% when compared to K-NN with black & white feature vector and SVM.

V. Conclusion and Future works

Observation:

Model	Accuracy
Blob Detection	90%
Black & White Ratio - A	85%
Black & White Ratio - B	65%
Linear SVM - SIFT	97.65%
RBF - SVM - SIFT	98.64%
Linear SVM - B&W	89.16%
RBF - SVM - B&W	95.27%
KNN – SIFT	99.35%
KNN - B&W	97.99%

From the above table, the following can be inferred – Accuracy KNN > SVM > B&W > Blob detection.

Using various techniques, we detected and counted the number of vehicles in each frame of the video. Each method resulted in better accuracy thus enabling us to choose an appropriate model to go forward with counting vehicles in a video.

This system can be extended to multiple parking lots as well as detect crime and unusual activity thus making the community safer.

References

[1] https://drive.google.com/drive/folders/0B4muPJH7ZVoVaFJ2ellzV3hrWG8

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

[3] https://www.mathworks.com/help/stats/classificationknn-class.html

[4] https://www.mathworks.com/help/images/examples/detecting-cars-in-a-video-of-traffic.html

[5] http://ieeexplore.ieee.org/document/4635804/

[6] https://www.mathworks.com/help/vision/ref/vision.foregrounddetector-class.html?searchHighlight=foreground%20detect&s_tid=doc_srchtitle

[7] https://www.mathworks.com/help/vision/ref/blobanalysis.html

[8] https://www.mathworks.com/help/vision/ref/vision.foregrounddetector-class.html