Technische Universität Berlin

- Institute for Open Distributed Systems –
Faculty IV

DCAITI Project Documentation
Modul SoSe 20/21

Detection of dynamic free space for parking lot occupancy with OpenCV

Jing Gu
Matrikel: 408463

Supervised by
Witzke Marcus

Declaration of Authorship

I, Jing Gu, declare that this thesis titled, etection of dynamic free space for parking lot occupancy with OpenCV and the work presented in it are my own. I confirm that:

This work was done wholly or mainly while in candidature for a research degree at this University.

Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.

Where I have consulted the published work of others, this is always clearly attributed.

Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.

I have acknowledged all main sources of help.

Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Date | Signed

Contents

1 Introduction

Two Cameras with three Lenses are installed on the top floor of the TC building and used to determine the occupancy status of the parking spaces in the viewing area.

In the past semester, other teams were attempted to record the occupancy status by detecting vehicles. The reverse of this semester is to attempt to determine the open spaces as precisely as possible from the given perspective and thus to determine the open spaces. The feature goal is combining these two methods to get a reliable result and the availability of possible parking spaces can be called upon the go.

There are several challenges. The first is Perspective, however in my project focus is the central Camera, perspective is not the main problem. The Second challenge is that dynamic Parking Space and parking area are specified by markings are needed to detect. The third challenge is to determine the number of cars or motorcycles can be parked in that parking area.

In this project, I use an unsupervised Machine Learning algorithm K-Means to cluster dynamic free space. unsupervised Machine Learning algorithm unlike classification (known as supervised learning), no a priori labeling of some patterns is available to use in categorizing others and inferring the cluster structure of the whole data. [1]For Parking Space with lines, a shallow neural network does a good job. In my work, the final result is the area of dynamic free parking space and coordination of the free space. However, it still not Robust for the changing light.

2 K-Means and Fully connected Neural Network

2.1 The K-means clustering algorithm

Expectation step: Compute the set of points associated to each centroid [2]

$$\forall\ 1 \leq k \leq K :\ C(k) \leftarrow \{i\ :\ k = \arg\min_{k} \|x_i - c_k\|^2\} \tag{2.1}$$

K : The number of clusters.
m : The number of Samples .
x_i : The ith sample in m samples.
c_k : The kth centroid.

Minimization step: Recompute the centroid as a the (weighted) mean of the associated data points [3]

$$\forall\ 1 \leq k \leq K :\ c_k \leftarrow \frac{\sum_{i \in C(k)} x_i}{\sum_{i \in C(k)}} \tag{2.2}$$

Cost Function:

$$J(c_1, ..., c_k) = \frac{\sum_i \min_k \|x_i - c_k\|^2}{\sum_i} \tag{2.3}$$

Repeat Expectation Step and Minimization step until the change of Cost Function within a threshold.

2.2 Fully connected Neural Network

Fully Connected layers in neural networks are those layers where all the inputs from one layer are connected to every activation unit of the next layer.

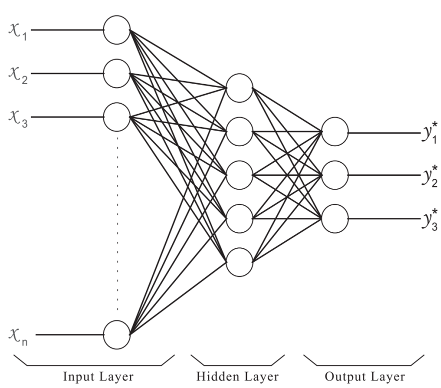

Figure 2.1: Fully connected Neural Network [4]

Input Layer: The input layer receives feature vectors.
Output Layer: The output layer gives prediction vectors.
Hidden Layer: The hidden layer processes and transmits the input information to the output layer [4]

The output of each neuron is described as follows:

$$y_j = f(\sum w_{ji} x_i) \tag{2.4}$$

f is the activation function can be a simple threshold, sigmoid, or hyperbolic tangent function. [4]

The sum of squared differences between the desired and actual values of the output neurons E is defined as follows:

$$E = \frac{1}{2} \sum_j (y_{dj} - y_j)^2 \tag{2.5}$$

y_{dj} is the desired value of output neuron j.
y_j is the actual output of that neuron j.
Backpropagation: The backpropagation algorithm is widely used as a primary part of a Neural Network model to minimize the value E. Activation Function here is sigmoid. [5]

$$\frac{\partial E}{\partial y_i} = \frac{1}{2} \cdot 2(y_{dj} - y_j) \cdot (-1) + 0, ..., 0 \tag{2.6}$$

$$\frac{\partial y_j}{\partial \sum w_{ji} x_i} = y_j(1 - y_j) \tag{2.7}$$

$$\frac{\partial \sum w_{ji} x_i}{\partial w_{ji}} = 1 \cdot x_i \cdot w_{ji}^{(1-1)} + 0, ..., 0 \tag{2.8}$$

$$\frac{\partial E}{\partial w_{ji}} = (y_j - y_{dj}) \cdot y_j(1 - y_j) \cdot x_i \tag{2.9}$$

Update Weights: To decrease the error, we then subtract this value from the current weight.

$$w_{ji}^{+} = w_{ji} - theta \cdot \frac{\partial E}{\partial w_{ji}} \tag{2.10}$$

3 Feasibility Analysis

3.1 Feasibility of K-Means

Images have been digitized with 8 bit/pixel 1280x720x3 pixel image format. Dynamic Free-space is the focus of this project. I do a manual cut of the image.

Figure 3.1: Cut of the Image

Change the Color Space of the image from RGB to HSV. Because in RGB color space it's hard to separate the cars from free parking space. And the RGB color space will strongly sensitive to the change of light.

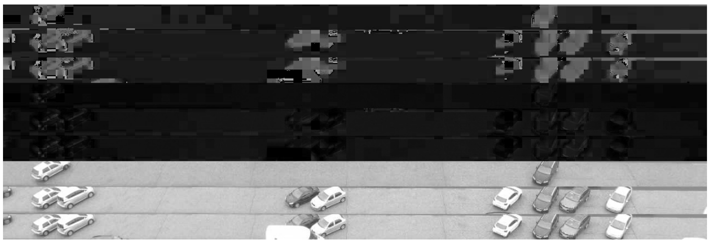

Figure 3.2: HSV Color Space

Because in the H channel, cars and free space are in different distributions,i choose H channel as processing Object.

Figure 3.3: H Channel

I flatten the H channel and map the flattened H channel in a 2d coordinate system. Figure 3.4 shows the distribution of cars and free space, the value in Figure 3.4 close to 25 is free space, and the value from 80 to 125 belongs to the car.

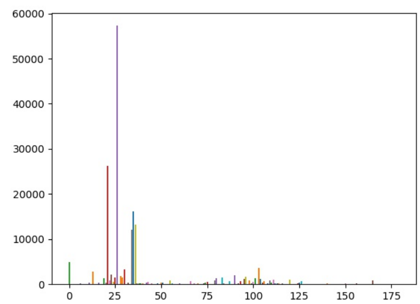

Figure 3.4: Frequence of Hue

The distribution of cars and free space approximates Gaussian distribution. I can use K-means to cluster Gaussian distribution.

3.2 Feasibility of Fully connected Neural Network

The training data is 28x28 pixel image format. I divide the raw data into two classes. One is Occupied parking space Figure 3.5 and the other is Free parking space Figure 3.6.

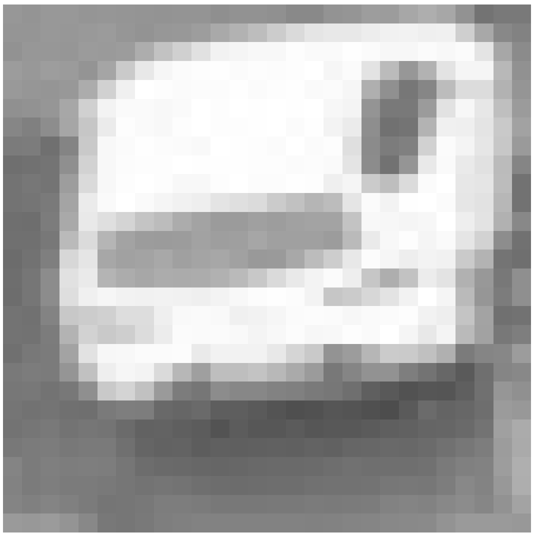

Figure 3.5: Occupied parking spaces

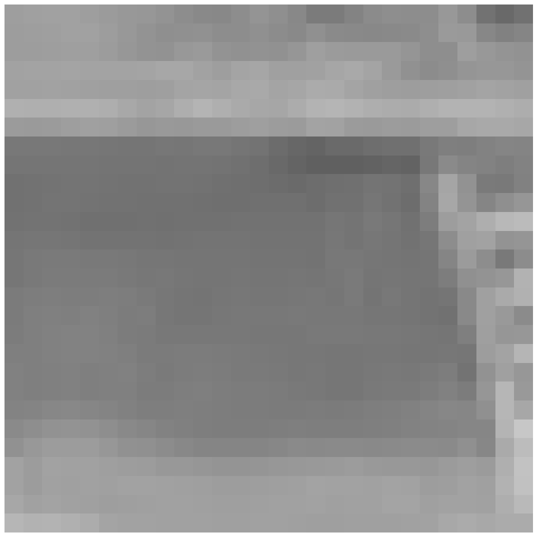

Figure 3.6: Free parking space

I draw the two classes into one Figure, the yellow points represent Occupied parking space and the blue points is the Free parking space, from the Figure 3.7, it's not difficult to find a boundary to separate these two categories. The Fully connected Neural Network is suitable at non-linear classification.

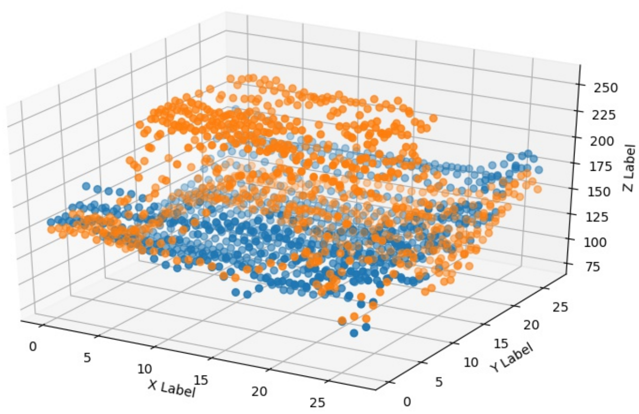

Figure 3.7: Distribution of Free space and Cars.

4 Details about Code

4.1 Hyperparameters of K-means

In OpenCV has already the K-means function,it has four hyperparameters:
Nclusters(K): Number of clusters required at the end. It is easy to choose, in Figure 3.4 and Section 3 have shown, there are two main clusters, one is the car and the other is free parking space. Choosing k equal 2 is reasonable.
Criteria: It is the iteration termination criteria. When this criterion is satisfied, the algorithm iteration stops. After a few attempts, I selected 10 times of iterations and accuracy of epsilon equal 1.0.
Attempts: Flag to specify the number of times the algorithm is executed using different initial labeling. The algorithm returns the labels that yield the best compactness.
flags: This flag is used to specify how initial centers are taken. Use the random centers.

4.2 Hyperparameters of Fully connected Neural Network

```
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                multiple                  100480
_________________________________________________________________
dense_1 (Dense)              multiple                  6450
_________________________________________________________________
dense_2 (Dense)              multiple                  2550
_________________________________________________________________
dense_3 (Dense)              multiple                  102
=================================================================
Total params: 109,582
Trainable params: 109,582
Non-trainable params: 0
```

Figure 4.1: Structure of Fully connected Layers

The input layer has 128 neurons, the first hidden layer has 50 neurons, the second hidden layer has 50 neurons, the output layer has 2 neurons correspond to two categories. The activation is Relu.
In Figure 4.2, the training accuracy and test accuracy increase with the epoch. There is no evidence shows over fitting.

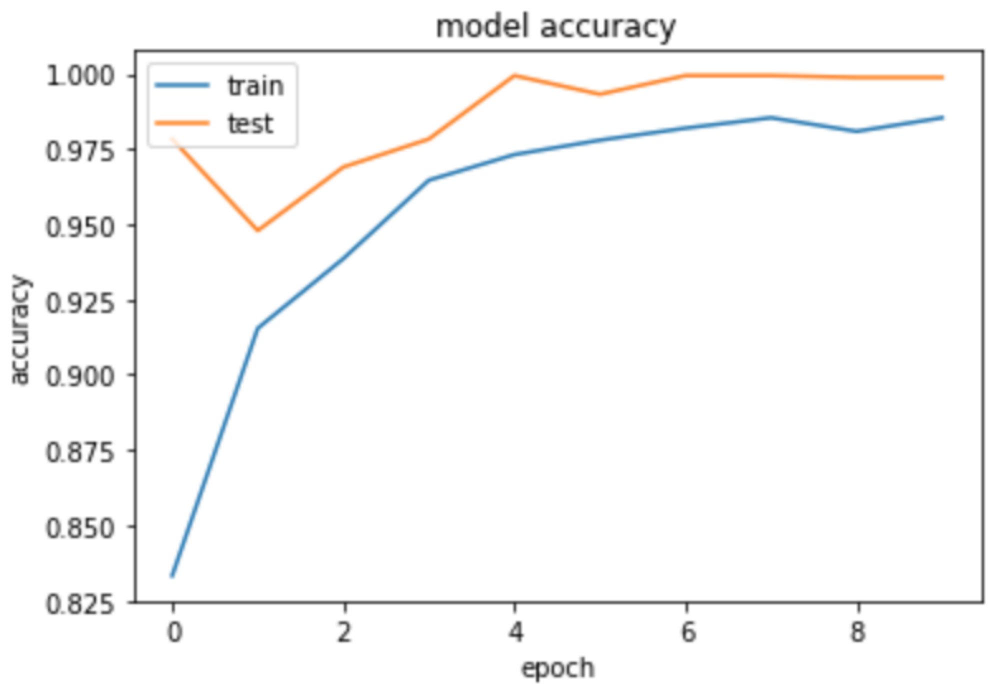

Figure 4.2: Training and Test Accuracy

4.3 Car sideline detection

After several Open and Close operations in Opencv, I got a mask of free space. I only choose the middle part of the mask to avoid the effect of traffic.
The green line intersects the white part in the mask, use the pixels beside the intersect point to distinguish the lines in the right of the car or on the left.

Figure 4.3: Mask of free parking Space

Use the sideline of cars, it's easy to figure out the area and location of the free parking space. Figure 4.4 the final output of dynamic free space.

Figure 4.4: Mask of free parking Space

4.4 About the Hard code

Hard code in main.py

1. Line 22: Set a update frequency of detection results.
2. Line 66 to 68: Set the location of free parking space in the middle manually .
3. Line 76: Set the location of putting text.

Hard code in kmeans.py

1. Line 14: The same function as Line 66 to 68 in main python file,set the location of free parking space in the middle manually .
2. Line 19: Parameter of Gaussian filter .
3. Line 44: Set threshold for Hue.
4. Line 68: Parameter of Canny function.
5. Line 71: Choose one line from mask to distinguish different lines, such as the line in the right of cars and the lines in the left of cars.
6. Line 83: If the distance between two lines small than 10 pixel delete the line has a small index.

4.5 Output of Terminal

The Output of the Terminal includes two main parts. The first part is the line position, it will be used to draw the dynamic free parking space. The second part is the time, record calculation time used in K-means and Neural Network. Because of the update frequency(Section 4.4), every 20 frames will update once the line list.

```
leftline  [44, 507, 891, 962, 1097]
rightline  [25, 166, 612, 944, 1069, 1144]
used: 0.21160578727722168
used: 0.0006968975067138672
used: 0.0015370845794677734
used: 0.0006971359252929688
used: 0.0006210803985595703
used: 0.002328157424926758
used: 0.0010228157043457031
```

Figure 4.5: Output of Terminal

4.6 Use pictures to generate test videos

In file Utilities has a python file called img2video.py, it is used to generate video from several continuous pictures. It has two parameters, that need to be set. One of them is the number of pictures, it means how many pictures will be used to generate a video. The second is the order of pictures, in another word, rename the picture chronologically.

5 Conclusion

5.1 Result and Summary

The First task is Segmentation of lane and parking space. The Second Task is the detection of dynamic combined free space. The third task is to calculate possible parking lots. The fourth task is Mapping of free space to virtual parking lots. In short in this project, dynamic Parking Space needs to be found and the area of it should be calculated. The final output is Geo-coordinates of dynamic Parking Space and the number of cars or motorcycles can be parked in that parking area.

The first three tasks have been roughly completed, detected dynamic free space, and parking area is specified by markings that are visualized in the output image. However, changing weather and different Lighting conditions result in unstable detection. Because of the unstable output, Geo-coordinates and the number of cars are also not accurate. In my work, I only output the local coordinate in the image. In further, when a stable output is possible, output the Geo-coordinate is meaningful.

Use K-means for dynamic free space detection in this project under some conditions is possible. Fully connected Neural Network has a really good performance at classifying the parking areas with specified lines.

5.2 Next to do

1. The most important task in the next is Robust. In K-Means, it uses Hue value for clustering. However, the features, such as texture and edges are not used in my work. In the next step try to combine the advance of Cluster algorithm and Deep Convolutional Network [6].

2. The second is to reduce the hard code parts and promote the method into the other two side cameras. It is similar to the first part in section Next to do, because an end to end solution treats the problem as a whole. After building the deep learning model, it is easy to use. [7]

3. After achieving the accuracy goal(Error is within 0.5 meters), then can focus on translating the pixel coordinates to Geo-coordinates.

List of Figures

References

[1] Sotiris Kotsiantis and Panayiotis Pintelas. Recent advances in clustering: A brief survey. *WSEAS Transactions on Information Science and Applications*, 1(1):73–81, 2004.

[2] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[3] K Krishna and M Narasimha Murty. Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439, 1999.

[4] Umut Orhan, Mahmut Hekim, and Mahmut Ozer. Eeg signals classification using the k-means clustering and a multilayer perceptron neural network model. *Expert Systems with Applications*, 38(10):13475–13481, 2011.

[5] Raul Rojas. The backpropagation algorithm. In *Neural networks*, pages 149–182. Springer, 1996.

[6] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1742–1750, 2015.

[7] Tobias Glasmachers. Limits of end-to-end learning. *arXiv preprint arXiv:1704.08305*, 2017.