**International Institute of Information Technology**

**Bangalore**

****

**Jan 2017 – May 2017**

**Machine Perception**

**Empty Car Parking Slot Detection**

**Submitted By: Group 36**

**Varun Jaiswal MT2016150**

**Sourabh Patel MT2016159**

**Vishal Bhurangi MT2016154**

**Shivam Agarwal MT2016127**

**Siddharth Dewangan MT2016133**

**Submitted To :**

**Dr. Dinesh Babu J**

**Project Report**

**Project Topic :** Empty Car Parking Slot Detection

**Task :**

Empty Car Parking slot detection using template matching, color detection technique and Haar Cascade.

**DataSet :**

We have generated **cars.xml** using OpenCV haartraining which is used for car detection.

For making cars.xml we have used :

* 1000 positive car images.
* 500 negative(background) images.
* 200 natural test images.

We have used 50 different parking slot images as input to our program for detecting the cars and empty parking slots in them.

**Assumption:**

1. We consider those images that are nearly the top view of the parking slot, so that each and every car can be distinguished accurately.
2. We assume every parking slot has figure( for our case a blue star) in the centre of slot. So when car is not present in slot then the figure is visible because we are considering top view of parking area and otherwise the figure is hidden by the car parked over it.
3. Cars and empty slots that are too far away in image, cannot be detected.

4. We consider good quality images (high illumination, good resolution).

5. On detection of car even though a particular car feature is detected, we consider it as a whole car is detected.

**Experiment :**

We have uset two methods in our program to detect car and empty parking slot.Firstly CascadeClassifier() is used to detect car images and bright color spot detection technique for detecting and counting the number of empty parking slots.

**Detecting car image :**

We have used cars.xml file (which we trained to contain car image features) as input to our program. Then CascadeClassifier() is used to read xml file. The detectMultiScale() is used to detect car images (with the output of CascadeClassifier which has the features of car) in given input parking slot image. After a car has been successfully detected we draw a rectangle over it.

**To detect empty parking slot we have used following techniques:**

**1. Template matching**

We have used cropped empty parking slot images to be used as template image to detect empty parking slots. For matching the template we have drawn a line on parking slots that can be detected when the slot is empty. matchTemplate() method is used for searching and finding the location of given template image(here empty car parking slot) in input image. When the template is matched we draw a rectangle around it to show empty parking slot.

There is no exact match for each and every empty parking slots, so some of the empty parking slots gets unmatched. Where as some of the empty parking slots gets detected multiple number of times, due to this we could not have exact number of empty parking slots.

So we used another technique for empty slot detection mentioned below.

**2. Bright color spot detection**

Here we have drawn bright blue color spots on the parking slots which are visible when no car is parked over them. The steps are:

a. We have loaded the image in gray scale and then blured it using Gaussianblur().

b. Now we threshold the image to reveal the light regions in the blurred image.Here we threshold colors in bright blue color range.

c. Now we perform a series of erosions and dialation to remove small blobs of noise from the thresholded image.

d. Now we count the bright spots in the image which is actually the count of the empty parking slots.

**Results :**

**Template matching performance:**

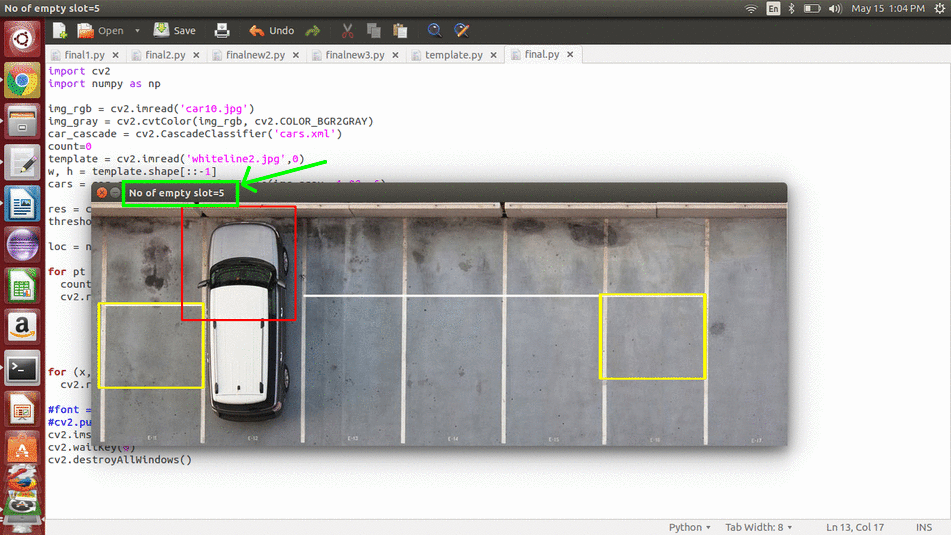
**1**. We changed parameters of matchTemplate() and analysed performance.

Different parameters are :

* cv2.TM\_CCOEFF
* cv2.TM\_CCOEFF\_NORMED
* cv2.TM\_CCORR
* cv2.TM\_CCORR\_NORMED
* cv2.TM\_SQDIFF
* cv2.TM\_SQDIFF\_NORMED

We found thatcv2.TM\_CCOEFF\_NORMED performed best because for the required threshold value, other parameters varied largely on varying threshold. This parameter provided the best matching results on the specific threshold value for different set of input images.

**2.** We found that the template matching technique detected the same empty parking slot multiple number of times for the best possible value of threshold. We have also found out that not all the empty parking slots are detected in the below image:



In the above image we can see that the total number of slots detected by template matching is 5 but actually it detected one of the slots multiple number of times. Hence the total accuracy for template matching for this image is :

Accuracy: 2/5\*100 = 40%.

Similarly we calculated accuracy for other images and found out that the average accuracy was approximately 45%.

**Bright color spot matching performance:**

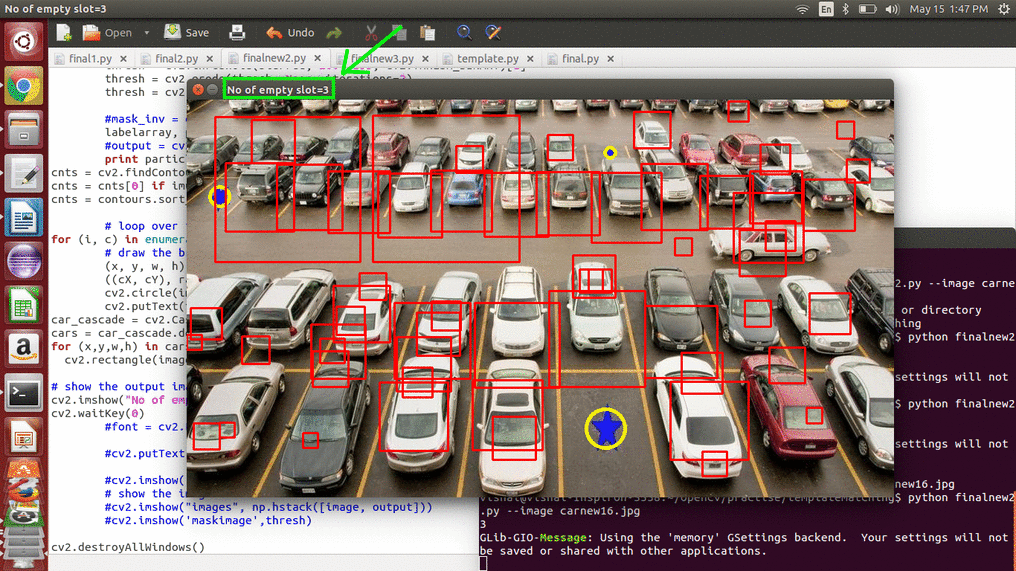
We found out that all the empty parking slots are detected by this technique. But in some cases few extra spots are also being detected that are not parking slots. In the below image following observation is made:

Accuracy: 3/3\*100 = 100%

We can say that all the 3 empty parking slot are detected.

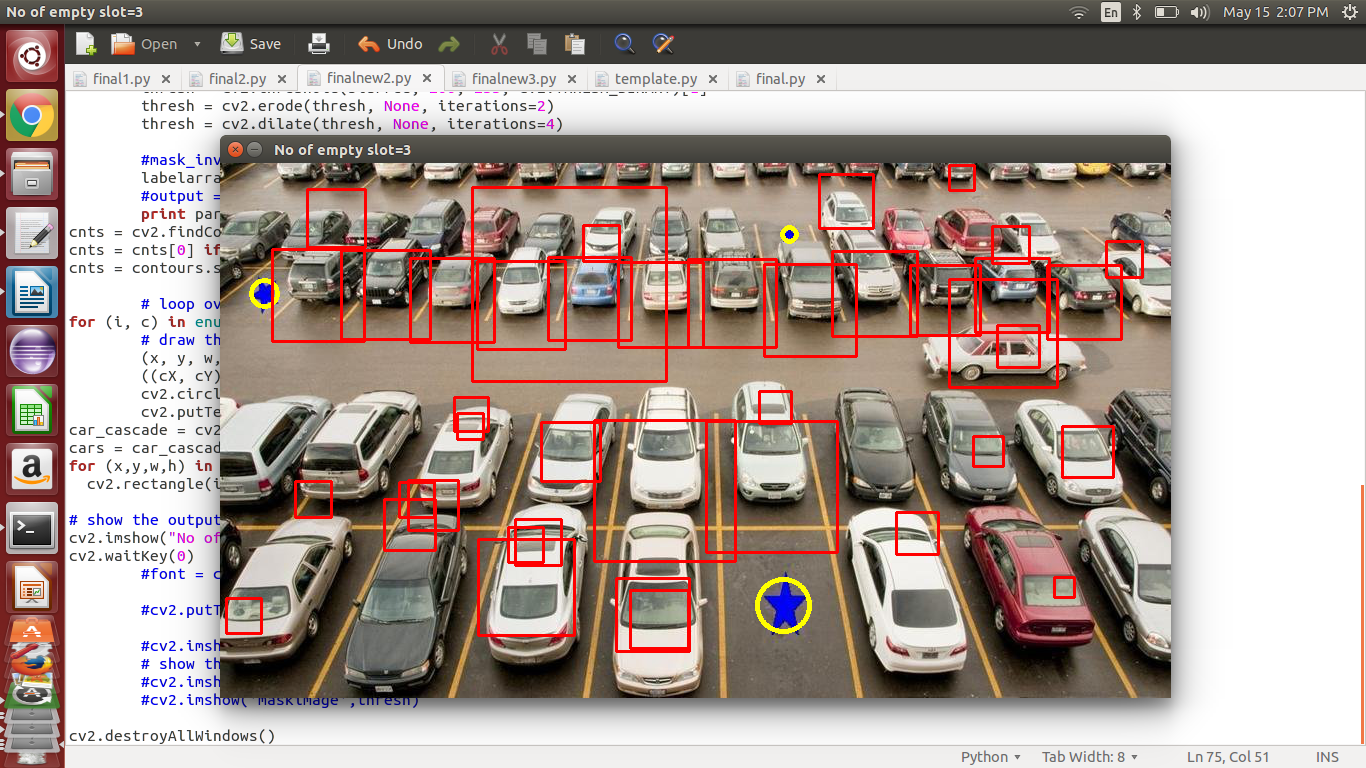
Similarly we calculated accuracy for other images and found out that the average accuracy was approximately 70%.

Also each empty parking slot is detected atmost once, hence it gives correct count of the empty parking slots.

****

The average accuracy of this technique for various images used is far more superior with respect to template matching.

**Car detection performance:**

****

In the above image out of 45 cars that are close enough the haar classifier detected 34 cars. So we can say that:

Accuracy: 34/45 \*100 = 75.5%

Similarly we calculated accuracy for other images and found out that the average accuracy was approximately 72.58%