Visual surveillance of on-street parking spaces

Răzvan-George Costea

June 2024

1 Problem description

In this project we have to implement a solution that analyse videos taken from a static surveillance camera and perform certain tasks like parking occupancy (task 1 and task 2), vehicle tracking (task 3) and counting the cars at the traffic light (task 4).

2 Data description

The context video are taken in different days or weeks with a few minutes before the testing data. For each task the input data is in a different form. For Task 1 the input is a frame from a video and a list which contains the indexes of parking spots that should be classified as occupied or. For Task 2 the input is a video and a state vector of the parking spots from the first frame. Task 3 and 4 have also a video as input, but Task 3 have a bounding box for the car that should be tracked.

3 Object detector

The object detector that we will use is YOLOv5 by Ultralytics pretrained from PyTorch Hub. This model can detect multiple object from an image among which cars and trucks.

4 Task 1 - Parking spaces occupancy

In this task we have to classify the parking spots given a list of indexes and an image. Because the camera is static the parking spot is located in the same region in every frame from every video. Therefore, we can look at this region of interest and crop it from the full image and feed the model. See figure 1.

YOLOv5 will output detections for every object in the scene, but we will consider only those detection with the name "car" or "truck". If the parking spot have a detection it means that this spot is occupied. We define a bounding box

Figure 1: Parking spots region of interest (ROI). In the left is the full frame with the ROI marked. In the right is the crop given as input to the object detection model.

vector $(predefined\_bbox_i)_{i=\overline{1,10}}$, where $predefined\_bbox_i$ is the bounding box for the i_{th} parking spot and that is sorted descending by the top-left corner. The predefined bounding boxes are taken using the model and an image that have all the spots occupied. Also, we define a state vector $(state_i)_{i=\overline{1,10}}$, where $state_i = 1$ if the i_{th} parking spot is occupied and $state_i = 0$ if it is not occupied.

To fill the state vector we need to match each detection with a predefined bounding box. A detection have a match with a predefined bounding box if the euclidian distance of the top-left corner from the detection and the top-left corner of the predefined bounding box is minimum. Also, the distance should be less than a specified threshold to not include cars that are on the nearby lane. See figure 2.

To get the output for Task 1 we run the procedure for the image and get the specified indexes from the returned state vector.

5 Task 2 - State of the parking places at the end of the video

For this task we need to output the state vector described in Task 1 for the last frame of the video. Because we already have the implementation from Task 1 we just iterate through all the frame and run the procedure for the last frame. This time we output the entire state vector

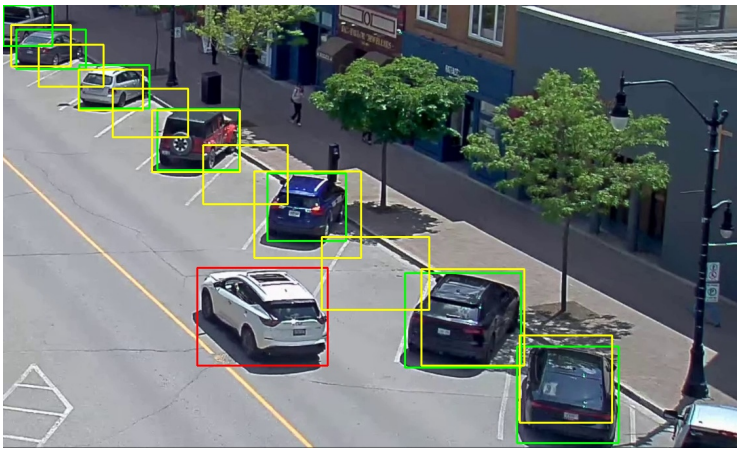

Figure 2: The crop with the bounding boxes drawn. The yellow boxes are the predefined ones, the red boxes are the car detections and the green boxes are the car detections that have been matched with a predefined bounding box. Note that the green ones were red before the matching algorithm

6 Task 3 - Vehicle tracking

For this task we have a video as input and the first frame bounding box for a car that should be tracked. The output is an annotated video and a file containing all the bounding boxes.

We iterate through all the frames from the video and for each one of them we detect all the cars and trucks. The detected bounding box that is drawn to the frame is the one that have the minimum euclidian distance from the last frame bounding box (see figure 3). In this way we keep track of the same vehicle the entire video. This procedure outputs the drawn frame and the bounding box that will be used for the next frame. The frame is written to a video stream that is saved on the disk with the required text file.

7 Task 4 - Number of vehicles in queue

For this task we have a video as input and we need to look again at a specific region from the frame to count all the cars (see figure 4). This time the region of interest is tighter than the parking spots so we need a mask to blacken all redundant pixels that could cause false positives (i.e. detected cars from the nearby lane). See figure 5 for masking procedure.

Using the masked region from the full image we detect all the cars and trucks for each frame. We compute the maximum number of cars that have been detected in the entire video and output this number in the text file.

Because the model could output multiple bounding boxes for the same car, we need to compute the IOU for each different detection pairs. If we have no

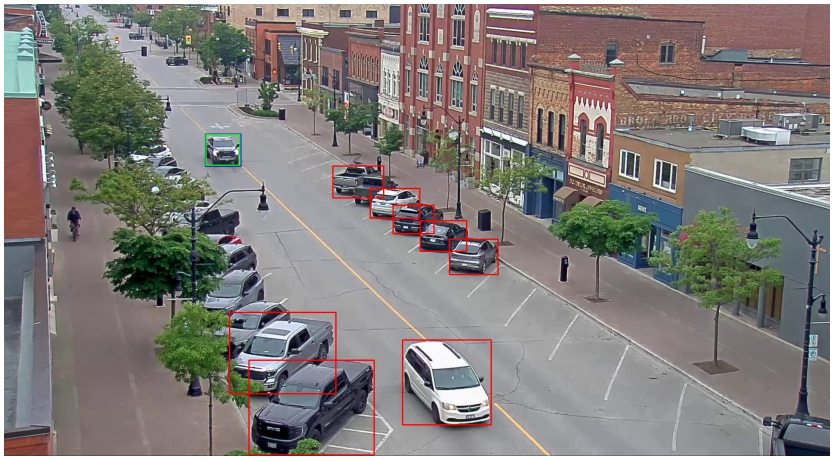

Figure 3: Image with all bounding boxes drawn. The red boxes are the cars detections that are not detections of specified car. The blue box is the given bounding box (i.e the bounding box of last frame). The green box is the detection of the car that we need to track, that is the most close to the blue one. Note that the green one was red (because is a car) before we chose the bounding box with the minimum distance to the blue one.

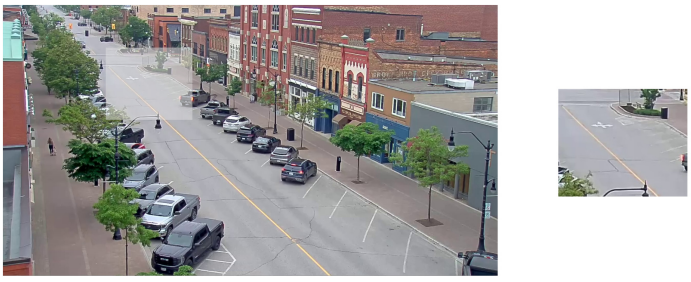

Figure 4: Traffic light region of interest (ROI). In the left is the full frame with the ROI marked. In the right is the crop.

Figure 5: The crop is multiplied element-wise with a binary mask resulting preserving only the lane where the cars will be counted.

duplicate detections the IOU should be 0.0 for all pairs. If we find a pair where the IOU is non-zero, than we exclude one of the detection from the counting.

8 Performance

The performance of the program on each task is listed in the bellow table. For the first three tasks the accuracy given the ground truth is close to 100% (note that for task 3 the ground truth provided is only for the first video; the program does not perform with 100% accuracy). For the forth task the drop in performance is due to poor counting. the model outputs multiple detections for the same car and some cars that are passing on the nearby lane may be counted, because they rooftop is seen in the masked region.

Task	1	2	3	4
Accuracy	97.6%	93.3%	100%	20%
GT samples	50	15	1	15
Train samples	50	15	15	15