COMPUTER VISION

Analysis of parking lot occupancy

Final project - September 2024

Introduction

Many video surveillance systems rely on computer vision algorithms to identify and track objects of interest in input videos and generate semantic information. An interesting example of this is the use of image processing techniques to automate parking analysis, detecting vehicle occupancy in real time and ensuring accurate information on parking availability. The information obtained from such systems can then be used to monitor parking occupancy over time or identify possible cars to be fined for incorrect use of the spaces provided.

The goal of this project is to develop a computer vision system for parking lot management, capable of detecting vehicles and available parking spaces by analyzing images of parking lots extracted from surveillance camera frames. The computer vision system to be developed must provide high-level information about the status of the parking lot (i.e., available and occupied parking spaces) for each frame of the provided sequences; this high level information should be displayed as a 2D top-view minimap, as shown in Figure 1.

The images to be used for system development include images of the empty parking lot and images of the parking lot with cars. The images of the empty parking lot are provided for the purpose of detecting the various parking spaces in a simplified scenario (i.e. without occlusions due to cars), while the images of the parking lot with cars shows the parking lot usage at various times of the day to be monitored. In more detail, the system to be developed should be able to:

1. Recognize and localize all parking spaces in the images of the empty parking lot, using a rotated bounding box aligned with the main dimensions of the parking space;
2. Classify all the parking spaces detected in point 1 according to their occupancy (0-“empty space”, 1-“occupied space”);
3. Segment the cars in the image into the following categories: 1-cars correctly parked (inside one of the parking spaces identified in point 1), 2-cars “out of place” (parked outside the proper locations);
4. Represent the current status of the parking lot in a 2D top-view visualization map, to be updated at each new frame with the current situation of free and occupied parking spaces.

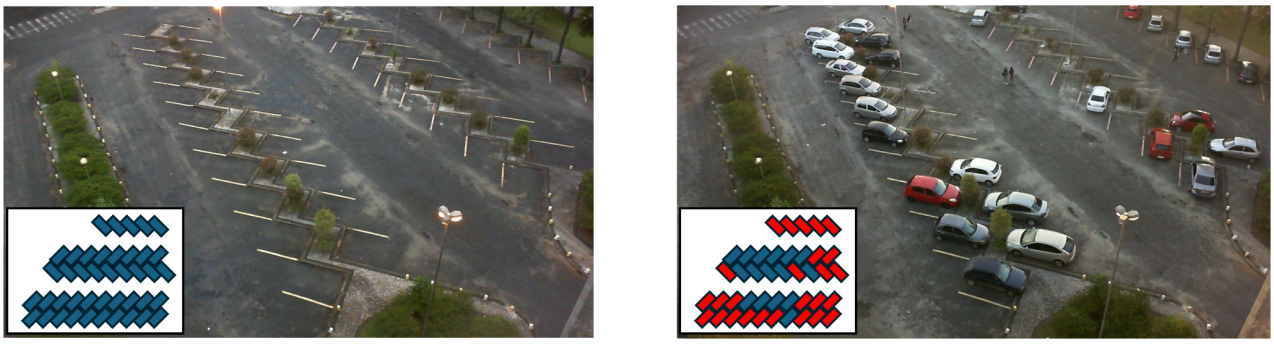

Figure 1: Example of the system output for a few frames of an input sequence. The minimap shows each parking space with a different color based on its occupancy: available (blue) or occupied (red).

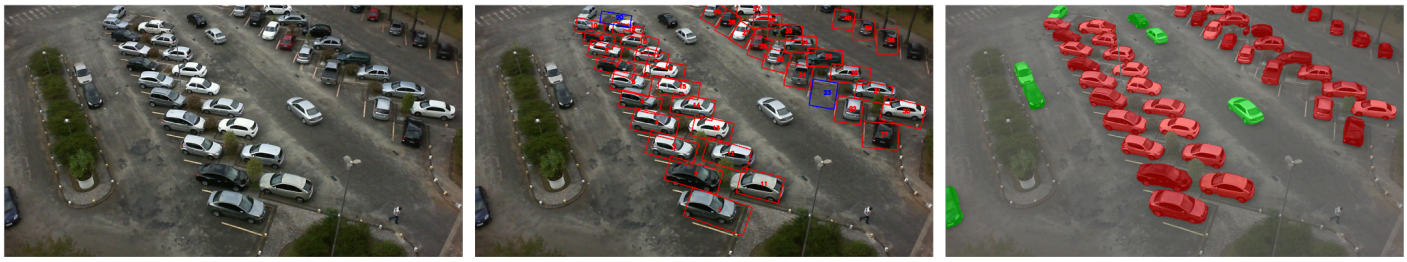

Figure 2: Example of the system outputs. From left to right: i) input image from a sequence; ii) parking space localization using rotated bounding boxes and different colors based on parking occupancy (blue empty, red occupied); iii) car segmentation, considering different colors for cars correctly parked (red) and cars outside the parking spaces (green).

As case studies for the development of the required system, various image sequences extracted from a surveillance camera are considered, differing in (i) the weather conditions, (ii) the time of day, (iii) the number of cars in the parking lot, (iv) the number of pedestrians in the parking lot. The system to be developed should be robust to all such conditions. In particular, it should recognize all the main elements (parking spaces and cars) albeit with different illumination conditions, and it should ignore any pedestrian in the parking lot.

In developing the project, the following simplifications can be made:

- you can assume that the pose of the camera does not change during each sequence;
- you can neglect the cars and parking spots in the upper-right part of the images, focusing only on the two main groups of parking spaces in the parking lot (i.e., the ones with the larger number of parking spaces).

To assess the robustness of your system, a benchmark dataset with annotations is provided at the following link:

https://drive.google.com/drive/folders/1CuVpazqMnap4khqNXaGU5q6N3K3Si9o1?usp=sharing

The benchmark dataset consists of 5 different sequences of a surveillance camera monitoring a parking lot, selected from the public available "PKLot" dataset[1].

For each sequence, five frames were selected corresponding to different times of the day and parking occupancy. Such images have been annotated with rotated bounding boxes and segmentation masks, and will be used to evaluate the performance of the computer vision system developed. As classes of interests, the following categories and labels have been considered:

- for parking space detection: available space (label id = 0); occupied space (label id = 1)
- for car segmentation: background (label id = 0); car inside a parking space (label id = 1); car outside a parking space (label id = 2).

The benchmark dataset includes also an additional sequence (i.e., "sequence0") containing 5 images of the empty parking lot without cars (as in Figure 1, on the left), differing in illumination conditions. These images should be used to compute an initial estimate of the location of each parking space in a simplified scenario (i.e., without occlusions due to cars), to be refined later on the images of the other sequences. **The system to be developed must be able to detect the parking spaces in each of the empty parking lot images (sequence0), under different illumination conditions.**

[1] https://web.inf.ufpr.br/vri/databases/parking-lot-database/

Performance measurement

For measuring the system performance, you should have a look and understand the following metrics:

- The mean Average Precision (mAP) - https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/
- The mean Intersection over Union (mIoU) - https://www.jeremyjordan.me/evaluating-image-segmentation-models/

Such metrics shall be used to evaluate the parking occupancy analysis system as follows:

- For parking space localization, the mean Average Precision (mAP) calculated at IoU threshold 0.5;
- For car segmentation, the mean Intersection over Union (mIoU) metric, that is the average of the IoU computed for each class (background, car inside a parking space, car outside a parking space).

The metrics mentioned above need to compare the output of your system against the ground truth, namely what is considered "the truth" for each output image. Ground truth annotations are already included in the dataset provided for the final evaluation of your system, in a separate file for each frame of the dataset. **Please note that the ground truth cannot be used as an input of your algorithm**.

The ground truth is stored in a set of files, one file for each input image. The information representing the ground truth (e.g., the rectangles enclosing the parking spaces or the pixels belonging to each car in the image) is expressed in such files based on the following standard:

- For parking space detection, a XML file containing a list of all the parking spaces in an image according to the annotations also used in the official PKLot dataset; every space is represented by a set of XML properties, namely: a unique ID number ("id"), a value "occupied" describing if the space is occupied (=1) or not (=0), a rotated rectangle defined by 5 parameters [x, y, width, height, angle], where (x,y) are the rectangle center coordinates and width and height are the bounding box main dimensions; the 5th parameter is the angle, defined as stated in the Opencv documentation[2];

- For car segmentation: a grayscale mask where each pixel is assigned the corresponding category ID (background, car inside a parking space, car outside a parking space).

Feel free to add more test images and videos, taken from the internet or acquired using your camera. In case you use additional images for the development of your system, you must include those images and related annotations (or a link to them) in your final delivery. If you use additional images, you are free to define your own standard (which you should describe in the report). If you agree with other groups to share the ground truth collection, be sure to share the standard. The organization of the ground truth collection is completely free; if you wish, you can use the dedicated section in the moodle forum.

[2] https://docs.opencv.org/3.4/db/dd6/classcv_1_1RotatedRect.html

Project delivery

The project must be developed in C++ with the OpenCV library **only**. **The project cannot be developed using deep learning; in particular you are not allowed to rely on state-of-the-art object detector or people detector based on deep learning**.

You need to deliver your project including:

- All the source code (C++), where each source file must contain the name of the main group member developer - **one author per file is allowed**; you should check that **the code compiles on the Virtual Lab**, that is considered the official building environment;
- CMake configuration files (the use of CMake is mandatory);
- A report (no page limit) presenting your approach and **the performance measurement on the dataset provided and linked above**. You shall report **the metrics and the output images** for every element in the dataset (see details below);

In particular, **you should include in your report** both quantitative and qualitative results of the performance of your system for each video clip in the provided benchmark dataset:

- for localization and segmentation tasks, report the output images (rotated bounding boxes and segmentation masks) of your system on each frame of the provided test sequences (including "sequence0"), together with all the requested metrics values (mAP, mIoU) computed with respect to the provided ground truth;
- for 2D visualization, report the image of the 2D top-view representing the parking lot occupancy obtained for each frame of each sequence.

When delivering your project, you should **clearly identify** the contribution of each member in terms of ideas, implementation, tests and performance measurement. You can organize the work as you prefer: you are not forced to assign one specific step to each group member. Please also include **the number of working hours** per person in the report. This is needed for a monitoring on our side of the effort requested – the evaluation will not depend at all on the number of working hours, but on the quality of the result. **Any requested detail that is missing will result in a penalty.**

It is not allowed to include code that was not written by any of the group members - this includes ChatGPT & similar tools.