Introduction to Computer Vision. Final Project

Determining the accessibility of parking spaces in unfavorable conditions

Egor Cherepanov
Emil Alkin
Vo Ngoc Bich Uyen

Content

1. Problem description (including the expected value of CV)
2. Dataset description with image examples
3. CV methods we plan to use

Problem description

Drivers in metropolitan cities spend **a large amount of time** seeking parking and are paid a high amount in parking fines. Looking for a parking spot is a common (and often stressful) activity for many individuals; it consumes around **one million barrels of oil** every day and contributes to **air pollution.** The growth of automobile ownership has produced an imbalance between parking demand and availability.

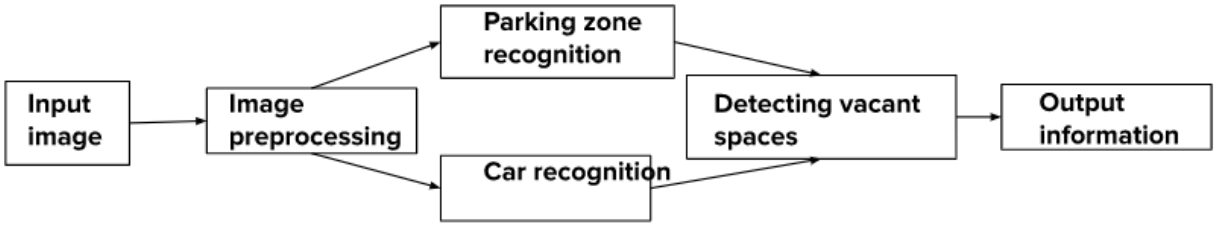
As a result, we suggested a project that uses **computer vision** and image processing to **locate vacant parking spaces** in parking lots based on bird's-eye view photographs

acquired by cameras in unfavorable conditions for recognition, i.e. parking lines are missing, covered with snow, etc.

Our computer vision algorithm is expected to generate an output, including a list of parking places, their center coordinates, and their occupation status in a certain geographic region, which we will then analyze to arrange cars in available parking spots.

Dataset description with image examples

In this project we want to create a program that takes a parking image as input (see examples below) and returns text information about available parking spaces.



1. **Input image**

The input image of our program is a photo of a parking lot in which the detection of parking lane markings is difficult / absent. View from above / from above at a slight angle.

2. **Image preprocessing**

Preparing the input image for further car/parking zone recognition.

3. **Recognition**

3.1. Car recognition

Then we want to use pre-trained models to detect cars.

We propose two different approach for this task:

- Pre-trained models for object detection. For example, we can use YOLOv5 model (https://pytorch.org/hub/ultralytics_yolov5/).
- Pre-trained models for segmentation. For example, we can use DeepLabV3 (https://pytorch.org/vision/stable/models/deeplabv3.html) or FCN (https://pytorch.org/vision/stable/models/fcn.html)

3.2. Parking zone recognition

If the marking lines are visible, we can use edge detection to identify potential parking spaces.

If there are none, we can use the following algorithm:

1. Recognize at least one car in perspective.
2. Through corners and straight lines estimate the approximate size of the parking lot and parking space.
3. Create an approximate parking grid and overlay it on the parking lot.
4. Get information about the remaining free spaces.

The picture above shows the perspective and conditions in which we want to implement our algorithm.

CV methods we plan to use

Preprocessing: filters, edge detection, counters detection, thresholding, projective transformation.

Algorithm: keypoints detection, bounding boxes, centroid detection, masking.

More examples of images: