Laboratory 5: Find a parking place!

Computer Vision 2022

P. Zanuttigh

CV 2022: HW / Projects

1. Introduction to C++ / OpenCV

No HW

2. **Histogram and Filtering**
3. Road Line Detection

- Select 1 among 2 and 3
- First «simple» HW: 3 pts
- Provide a very short report or comments in the source
- On/off mark

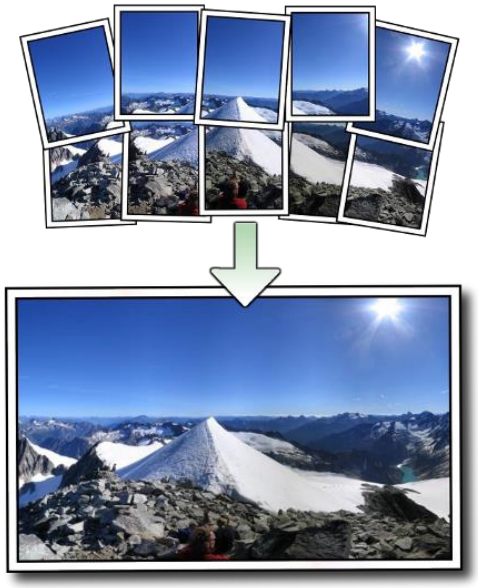
4. **Image Mosaicing**
5. **Parking Spaces Det.**

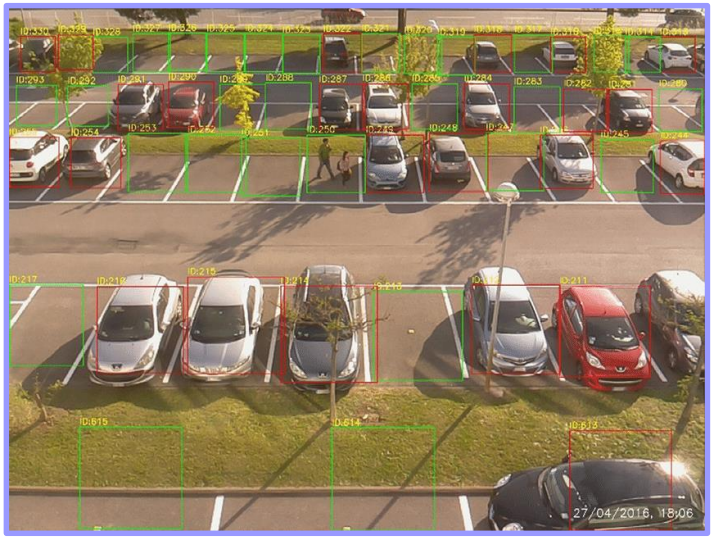
- Select 1 among 4 and 5
- Second «advanced» HW: 6 points
- More detailed report with results
- Mark based on solution quality

Final Exam (written in classroom): ~23 points

Final mark: 3 + 6 + 23 = max 32 points

CV 2022: Final Projects

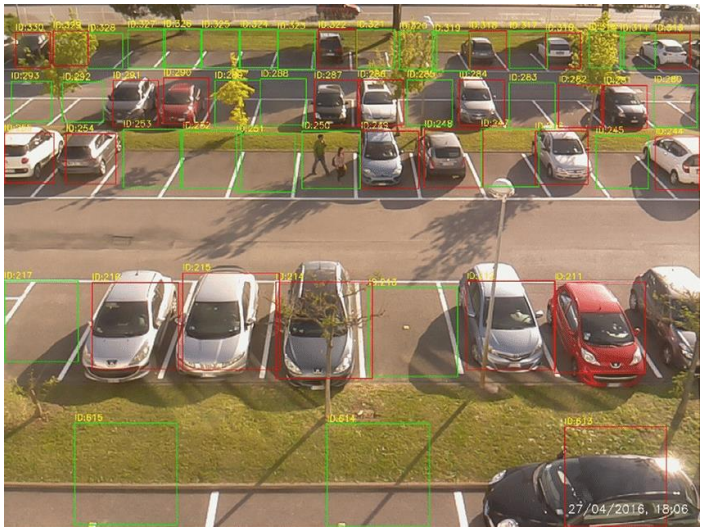




Two options:

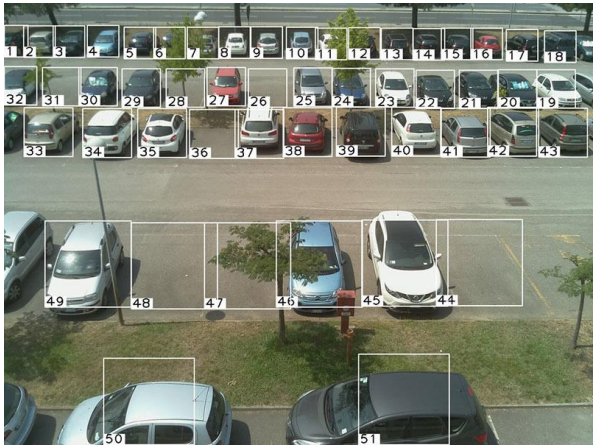
1. *Image mosaicing with feature descriptors (LAB4)*
2. ***Detection of free parking spaces (LAB5)***

Project 2: Parking Slot Detector

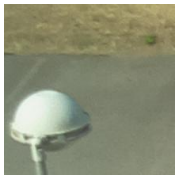


- Find free parking slots in the provided images
- Analyze each slot's region and detect if it contains a car
- Both "traditional" and machine learning based approaches can be used

The Datasets: CNRPark



Subset	Cams	Collection Period	Weather	Frames	Patches
CNRPark	2	July 2015 (2 days)		242	12,584
CNR-EXT	9	Nov. 2015 - Feb. 2016 (23 days)		4,081	144,965



- Download from http://cnrpark.it/
- Provides both the full images and the single slot patches
  - You can use the patches to train binary classifiers based on ML techniques
- The `.csv` files contains the location of the slots
- File `CNRpark+EXT.csv` and the `.txt` files contain the ground truth

The Datasets: PkLot

Parking1a

Parking1b

Parking2

- Download from https://web.inf.ufpr.br/vri/databases/parking-lot-database/
- Provides both the full images and the single slot patches
- You can use the patches to train binary classifiers based on ML
- It considers larger parking lots with more slots that can be also rotated in different ways
- The slot positions and the ground truth are inside the `.xml` files
  - Have a look at the OpenCV `FileStorage` class to open `.xml` files

Parking Slot Detector

1. Download the datasets and extract the information you need for the target task
2. The datasets are divided into train and test sets
   - The training set is useful if you use ML-based approaches
3. *Task:* find which parking slots are free and which are occupied
4. Possible approaches:
   - Run edge/corner/feature detector into the slot and analyse found features
   - Analyse image statistics into the slot
   - Extract features (e.g. SIFT, ORB) , then use ML classifiers, e.g., SVM or Random Forests
   - Deep learning binary classifier: you can segment the slots, normalize them to a common size and orientation and then feed to the DL classifier (e.g., a 3-4 layer CNN)
   - Deep learning object detector (e.g., look for cars or motorbikes)

Notes

- You can choose your preferred strategy, any idea different from the proposed ones is welcome
- Develop the approach in C++
  - In case of Deep Learning based approaches use Python only for the deep networks module but keep C++ for pre- and post-processing
- ***Avoid directly cutting and pasting material from the web***
  - in particular the cnrpark webpage provides a deep network already trained for the task, do not use it
  - If for any reason you use material from the web for side activity (e.g., load xml, visualization, pre-available deep nets…) declare it
  - Not declared material from the internet will result in a big penalty
- Recall you need to deliver the code, the report (mandatory for the final project) and some output results