Analysis of the student's activities in an University Campus

I. Introduction

One of the key tasks in computer vision and image processing is conducting trajectory analysis of moving objects within a given environment. Such an analysis is crucial, as it provides motion insights about the objects in the scene. Additionally, it allows for the collection of all trajectories over time, enabling a statistical characterization of the activities performed.

For example, in the case of vehicles, it is possible to gather traffic-related data and broadcast information regarding areas with heavy congestion. Once this information is available, traffic bottlenecks can be mitigated. If the application is focused on human activities, trajectory analysis can also be highly valuable. Specifically, it allows for the collection of typical movement patterns and the detection of abnormal or suspicious behavior, which can be interpreted as uncommon paths that deviate significantly from the usual trajectories.

Another category of applications can be considered. For instance, in mobile robotics, trajectory analysis can enhance Human-Robot Interaction systems, while in the automotive industry, it provides input for Advanced Driver Assistance Systems.

However, designing an algorithm for trajectory collection presents several challenges. These include the high variability in pedestrian movement, the impact of an individual's pose on their appearance in images, variations in clothing, atmospheric conditions affecting illumination, background clutter, and occlusion. All these factors contribute to making pedestrian detection a complex problem to solve.

The objective of this work is to develop an algorithm capable of detecting pedestrian locations to derive their trajectories. Conventional handcrafted features will be employed for this purpose.

As a final note, please keep in mind that the algorithm's output should aim to provide the richest possible visual information.

II. DATASETS

For this work we will use the publicly benchmark datasets. Among several datasets, we will use the PETS family dataset. The dataset is available on the GitHub of the course (Crowd-PETS directory). Here, a set of images - $\mathbf{S2.L1}$ is available and it is concerned with *sparse crowd*. This means that isolated pedestrians are considered with little interactions between them. This sequence has a level of difficulty of 1 (in a range of 1-3 complexity levels).

Considering the $\mathbf{S2.L1}$, there are several acquisitions, each containing a different view. The views are numbered as follows: $\mathrm{View001}$, $\mathrm{View002}$, ..., $\mathrm{View008}$.

In this work we will concentrate in the $\mathbf{View001}$ sequence that contains 795 frames.

The students, however, are free and welcome to use more difficult and challenging views sequences if they feel like. Of course, a reward will be granted.

Fig. 1 shows some images samples belonging to the dataset $\mathbf{S2.L1}$ in the $\mathbf{View001}$ sequence.

Figura 1. Frame samples from the dataset $\mathbf{S2.L1}$ in the $\mathbf{View001}$ sequence.

III. Ground Truth data to measure the Algorithm performance

To measure the detection performance of the algorithm, we need to have some gold standard, or ground truth (GT) of the pedestrians position. Basically, the idea is to compare the algorithm's predicted output with the GT locations. One way to perform this task is to use the bounding boxes to represent the location of the pedestrian in the image domain. Thus, comparing the bounding boxes of the GT against the ones obtained with the algorithm it is possible to ascertain if the estimated bounding boxes are close or not comparing to the GT positions.

The GT information is available in the "gt.txt" file available in the GitHub, concretely in the directory PETS-S2L1. The contents of the file "gt.txt" follows the same structure as in [1] (see also Table 2 in [1]). Each line of the file contains:

1) *Frame number*: Indicate at which frame the object is present
2) *Identity number*: Each pedestrian trajectory is identified by a unique ID
3) *Bounding box left*: Coordinate of the top-left corner of the pedestrian bounding box
4) *Bounding box top*: Coordinate of the top-left corner of the pedestrian bounding box
5) *Bounding box width*: Width in pixels of the pedestrian bounding box
6) *Bounding box height*: Height in pixels of the pedestrian bounding box
7) *Confidence score*: Indicates how confident the detector is that this instance is a pedestrian. For the ground truth and results, it acts as a flag whether the entry is to be considered.
8) *x*: 3D x position of the pedestrian in real-world coordinates (-1 if not available)
9) *y*: 3D y position of the pedestrian in real-world coordinates (-1 if not available)
10) *z*: 3D z position of the pedestrian in real-world coordinates (-1 if not available)

Fig. 2 shows an example of an image sample (left) and the same image with the corresponding GT detections represented in bounding boxes (right).

The work will have several goals, the majority intend to enrich the *visual information* that can be extracted from the image sequence. Thus, the students are welcome to fulfill the following challenges:

1) Plot the GT (readable from the gt.txt file) and draw the bounding boxes in each frame in the sequence, (see Fig. 2 right). **(3.0v)**
2) Now, using your detector algorithm, perform the tracking of pedestrians. The predicted bounding boxes should be visible for each detection. Assign a label (*i.e.*, a number) for each detected bounding box. At this stage is not required to have the same label assigned for a given pedestrian. Label switching can occur. **(4.0v)**

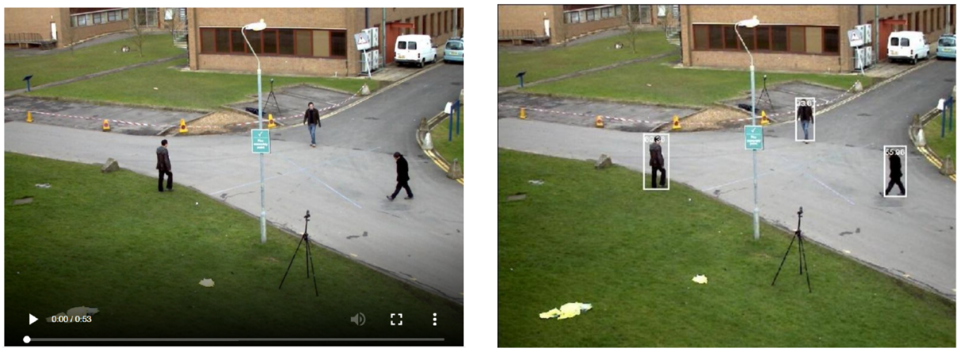

Figura 2. One frame sample from the sequence $\mathbf{S2.L1}$ in the $View001$ (left) and the same frame with the ground truth in bounding boxes (right).

3) Plot the performed trajectories. To avoid a possible excess of the information visualisation, you can plot the trajectories dynamically. **(4.0v)**
4) Provide consistent labels through time. This means that a given pedestrian should be assigned to the same label through the sequence. **(2.0v)**
5) Provide to the user, the information regarding the map (*i.e.*, occupancy) of the trajectories performed in the video. Specifically
   - Provide a heatmap, using a Gaussian distance metric (or other), where the color is assigned to the number of occurrences in a given position (region) of the image. Concerning this regard different heatmaps can be generated, this can include (i) static heatmap, (ii) dynamic heatmap. **(2.0v)**
6) Using the Expectation-Maximization (EM) algorithm, provide a statistical analysis concerning the trjectories performed by pedestrians. **(1.5v)**
7) Provide an evaluation performance of the algorithm. Specifically provide: (i) the success plot (see Sec. IV for details), and (ii) the percentage of False Negatives (FN) or misdetections and False Positives (FP). Also, provide figures illustrating the success plot, FPs and FNs, and also some frames illustrating the FPs and FNs. **(2.5v)**
8) Using a deep neural network, provide a comparison between your method and the one provided by the deep network (take a look at Pedestrian Detection link). **(1.0v)**

IV. Evaluation metrics

One important issue to be considered is that every algorithm hat its own limitations. This means that, no matter the approach is adopted, there is always some failures regarding the true location of the pedestrian. For instance, a merge or split in a given bounding box that can occur. Also, some misdetections may occur as well. Thus, one way to evaluate the algorithm is to use evaluation metrics. An evaluation strategy can be done as follows:

1) The first step is to built the ground truth as already mentioned.
2) After this stage, the students are in conditions to show both the ground truth and the estimated bounding boxes provided by the algorithm.
3) Now, evaluation must be done. To accomplish this, the following metric is suggested:
   - Provide the *success* plot using the Intersection over union measure (IoU) that is defined as follows:

     $$IoU = \frac{R_d \cap R_{gt}}{R_d \cup R_{gt}} \quad (1)$$

     where R_d is the detected region estimated by the algorithm and R_{gt} is the ground truth (manual labeled) region. Basically, the IoU provides a measure of the overlap (or match) between the R_d and R_{gt}. A score of $\text{IoU} = 1$ means a perfect match is obtained, and $\text{IoU} = 0$, means that the target is loss.

     The success plot shows the percentage of frames whose bounding box overlap ratio is higher than a given threshold. For threshold, it can be considered the values ranging from 0 to 1, with step of, *e.g.* 0.1.

Deadline: The students must upload the projects to my e-mail jacinto.nascimento@tecnico.ulisboa.pt until, April 6th, 23h59m. when submitting your project name your zip file simply with your group number, *e.g.* **14.zip**.

V. Reading material

The students are welcome to read the following paper:

[1] L. Leal-Taixe, A. Milan, I. Reid, S. Roth, and K. Schindler “MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking”, arXiv 2015.