University of Bucharest
Faculty of Mathematics and Computer Science
Department of Computer Science

Alexandra Diaconu
Ciprian Ceaușescu
Bogdan Alexe
MSC in Computer Science
June 2023

Computer Vision - Project 2

Visual traffic monitoring at a road intersection

Objective

The goal of this project is to develop an automatic system for video analysis that enables visual traffic monitoring at a road intersection (Figure 1). The system gets as input data the video stream from a static camera and should be able to: (i) classify road lanes as being occupied or not given a video frame; (ii) track a specific vehicle in a given video; (iii) count the trajectories of vehicles in the intersection in a given video.

Introduction

In this project you have to analyze video data obtained from a static camera with the desired goal of monitoring the traffic in an intersection. The videos are collected at different times during day/week so it is easy to observe that there are some changes in illumination in the scene specific to each video (Figure 1). For each of the three tasks that you have to tackle in this project you are given a "context" video that is taken with a few minutes be-

Figure 1: *The static camera acquires images of the intersection at different times in a day/week.*

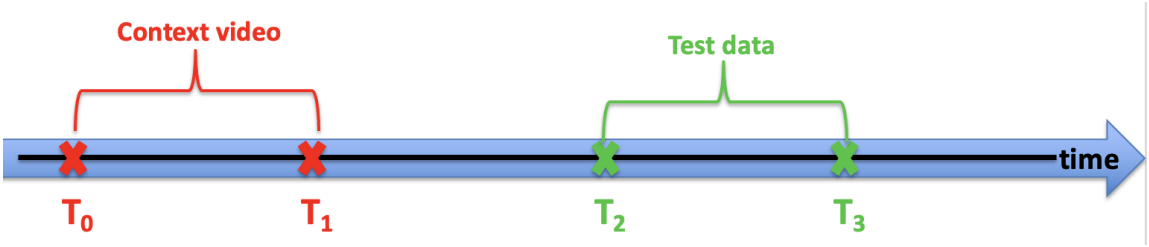

Figure 2: *The context video data is taken from time T_0 to time T_1, a few minutes before the test data is taken from time T_2 to time T_3.*

fore the test data (images or video) that you have to analyze was acquired using the same static camera (Figure 2).

Data description and grading scheme

The release data directory (available at https://tinyurl.com/CV-2023-Project2) contains three directories: *train*, *test* and *evaluation*. The directories *train* and *test* have similar structure, although the *test* data will be made available after the deadline. The *train* directory contains the data organized in four subdirectories corresponding to the three tasks that you need to solve and the context videos shared by these tasks. The subdirectories are:

- *context_videos_all_tasks* - this directory contains 15 training videos which represents the context video data for each of the three tasks. Notice that each video is taken at a given moment of a day from T_0 to T_1.

Figure 3: *Road lanes are numbered from 1 to 3 for the upper left part of the intersection, 4 to 6 for the right part and 7 to 9 for the bottom right part of the intersection. We do not consider the left part of the intersection.*

- *Task1* - this directory contains 50 training images (3 images corresponding to the first 10 training (context) videos and 4 images corresponding to the remaining 5 training videos - so in total there are $3 \times 10 + 4 \times 5 = 50$ training images) and 15 query txt files, each query file corresponding to a training video. Notice that for each training video, the corresponding images are taken from T_2 to T_3.

  The Task 1 consists in correctly classifying the road lanes listed in the query file as being occupied (label 1) or not (label 0). The lanes are numbered as in Figure 3, from 1 to 3 for the upper left part of the intersection, 4 to 6 for the right part and 7 to 9 for the bottom right part of the intersection. We do not consider the left part of the intersection.

  The format of the query files is the following: on the first row it is specified the number L of lanes for which you have to do the binary classification and then each of the following L rows lists a lane number. Please also refer to the examples presented in Lecture 14.

  Grading scheme for Task1. In the test scenario we will release 15 testing videos that will offer the context data. For Task 1 we will release 50 testing images, following the same distribution as in training ($3 \times 10 + 4 \times 5 = 50$ testing images). By correctly solving the Task1 on all images you will get 2 points. You get 0.04 points/image if your algorithm solves correctly the binary classification for all lanes specified in a query file.

- *Task2* - this directory contains 15 training videos taken in the interval from T_2 to T_3. The task is to track a specific vehicle from the initial frame to the final frame of the video (see Figure 4). The initial bounding box of the vehicle to be tracked is provided for the first frame (the annotation follows the format [xmin ymin xmax ymax], where (xmin,ymin) is the top left corner and (xmax,ymax) is the bottom right corner of the

Figure 4: *Examples of train videos for Task2: the figure displays four frames of a training video containing annotation (green bounding-box) of a specific vehicle.*

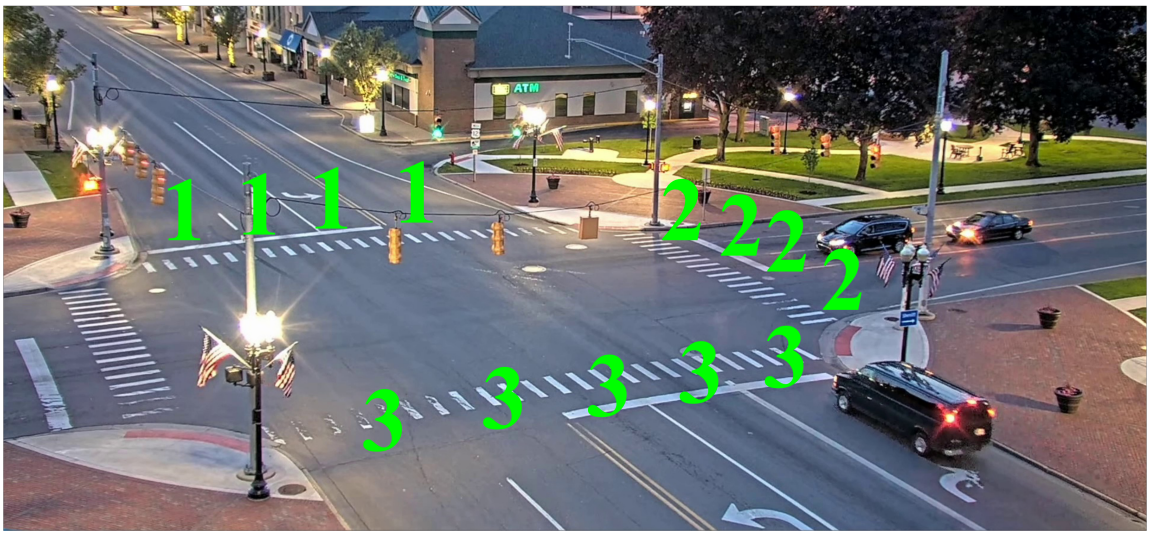

Figure 5: *Regions of interest for Task3 are numbered with 1 (upper part of the intersection), 2 (right part) and 3 (bottom part). We do not consider the left part of the intersection.*

initial bounding-box).

In each video we will consider that your algorithm *correctly tracks the vehicle* if in more (greater or equal) than 80% of the video frames your algorithm *correctly localizes the vehicle to be tracked*. We consider that your algorithm *correctly localizes the vehicle to be tracked* in a specific frame if the value of the IOU (intersection over union) between the window provided by your algorithm and the ground-truth window is more than 20%. The format that you need to follow is the one used in the ground-truth files (located in the subdirectory *ground-truth*) with the first line containing the number of frames N of the video, and each line having the format [frame_index xmin ymin xmax ymax]. The first frame for which we provide the bounding box initialization has frame_index 0, the last frame of a video with N frames has frame_index $N-1$. Please note that the first line of the annotation file has the format [N -1 -1 -1 -1] as it is easy to load an entire matrix $M \times 5$ (first line is [N -1 -1 -1 -1], then the following $M-1$ lines are of the form [frame_index xmin ymin xmax ymax], so $M \leq N+1$) to assess the correctness of your algorithm. Notice that if the vehicle disappears from the video your algorithm should not output any detection in the corresponding frames (in this case $M < N+1$).

Grading scheme for Task2. In the test scenario we will release 15 testing videos similar with the training videos. By correctly solving the Task2 on all videos you will get 1.5 points. You get 0.1 points/video if your algorithm correctly tracks the vehicle in a video.

- *Task3* - this directory contains 15 training videos. The goal here is to count the trajectories of vehicles in the given video. We are interest to count the number of vehicles between the three regions of interest (regions are numbered as in Figure 5). The ground-truth files specifies the number of vehicles that go from region 1 to region 2, from region 1 to region 3, from region 2 to region 3 and so on.

Grading scheme for Task3. In the test scenario we will release 15 testing videos for Task3. By correctly solving the Task3 on all videos you will get 1.5 points. You get 0.1 points/video if your algorithm correctly predicts the number of all vehicles for all possible combinations of regions in a given video.

- 0.5 points - ex officio. Please note that we will award the 0.5 points only to those students who submit their results in the REQUIRED format.

Evaluation

The directory *evaluation* shows how the evaluation will take place on the test data after the deadline. It contains the following subdirectories:

- *fake_test* - this directory exemplifies how the test data will be released in the *test* directory (similar with Project 1), keeping the structure of the previously described *train* directory;
- *submission_files* - this directory exemplifies the format of the results data that we expect from you to submit in the second stage. You will have to send your results in this format, uploading a zip archive of a folder similar with the one called *Alexe_Bogdan_407*. Please note that if you don't submit your results in this format you will not get the 0.5 points from ex officio;
- *code_evaluation* - this directory contains code that we will use to evaluate your results using the ground-truth data. Make sure that this code will run on your submitted files. The ground-truth data will be released after you send us your results.

Deadlines: Submit a zip archive containing your code, all auxiliary data that you are using (templates, models, etc.) and a pdf file describing your approach until Thursday, 22^{nd} of June using the following link https://tinyurl.com/CV-2023-PROJECT2-SUBMISSIONS. Please do not include in your archive training images or videos. Notice that this is a hard deadline, no projects will be accepted after the deadline. Your code should include a README file (see the example in the materials for this project) containing the following information: (i) the libraries required to run the project including the full version of each library; (ii) indications of how to run the solution for each task and where to look for the output file. Students are allowed to submit their solution in the format of Jupyter Notebook files with the code being commented. This can replace the pdf file describing their approach. Students who do not describe their approach (using comments throughout the code or a pdf file) will incur a penalty of 0.5 points.

On Friday 23^{rd} of June we will make available the test data. You will have to run your system on the test data provided by us and upload your results in the same day as a zip archive using the following link https://tinyurl.com/CV-2023-PROJECT2-RESULTS.