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

Alexandra Diaconu
Bogdan Alexe
MSC in Computer Science
June 2025

Computer Vision - Project 2

Beyond one view: intersection surveillance using multiple cameras

Objective

The goal of this project is to develop a fully automatic video analysis pipeline that enables multi-camera surveillance of a single road intersection. Given streams from multiple fixed cameras (Figure 1) that monitor the intersection, the system should perform the following tasks:

(1) *Temporal localisation* – find a 3-second query video from one camera (A) inside a 30-second reference video from another camera (B), and report the exact frame index at which the query begins in camera B's timeline;

(2) *Cross-view single vehicle tracking* – given a single tight bounding box around one vehicle in one camera (A), track this vehicle through the synchronised streams of cameras A and another camera (B) despite possible changes in viewpoint, scale, and occlusion of the vehicle;

(3) *Directional traffic counting* – given two overlapped streams from two cameras (A and B), count every vehicle that moves from a user-defined origin direction to a destination direction across the two videos, producing per-direction flow statistics.

Introduction

Urban intersections are the beating heart of any city's road network, yet they remain bottlenecks for safety and efficiency. Traffic management centres increasingly deploy

Figure 1: *Simultaneous snapshots of the intersection scene captured by all three cameras.*

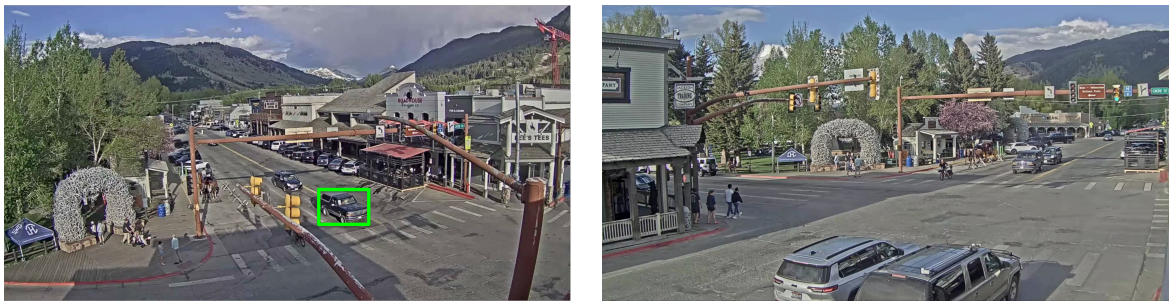

Figure 2: *Illustration of* ***Task 2*** *— cross-view tracking of a single vehicle. A tight bounding box marks the target in the first frame of camera A; the same vehicle should be tracked throughout both synchronised videos.*

multiple inexpensive CCTV cameras with overlapping but non-identical views to gain richer situational awareness. However, raw footage alone cannot provide actionable insights; it must first be synchronised, conceptually stitched, and then analyzed.

In this project you will assume the role of a smart city computer vision engineer. Our data contains footage from three fixed surveillance cameras that overlook a busy urban intersection. For each of the three tasks you will receive a *pair* of videos recorded by two of these cameras, hereafter denoted *camera A* and *camera B*. Because the videos were captured at different times of day and on different days of the week, each pair exhibits noticeable variations in illumination and traffic density, adding a realistic layer of difficulty to the analysis.

Data description and grading scheme

The release data directory (available at `https://tinyurl.com/CV-2025-Project2`) contains three directories: `train`, `test` and `evaluation`. The directories `train` and `test` have a similar structure, although the test data will be made available after the deadline. The `train` directory contains the data organized in three subdirectories corresponding to the three tasks that you need to solve. The subdirectories are:

- **Task 1** – *Temporal localisation*. This folder contains 15 training pairs. Each pair consists of a 3-second *query* video, recorded by camera A, and a 30-second *reference* video, recorded by camera B. The query video appears entirely inside the reference video; your task is to return the frame index of the reference video taken with camera B at which the first frame of the query begins. Notice that the first frame in each video has index 0.

  Grading scheme for Task1. In the test scenario, we will release 15 query/reference pairs in the same format. You receive 0.10 points for every pair in which your predicted start frame is within ± 60 frames of the ground-truth location.

- **Task 2** – *Cross-view single vehicle tracking*. This directory holds 15 training pairs, each pair consisting of two videos recorded by different cameras (cameras A and B). The

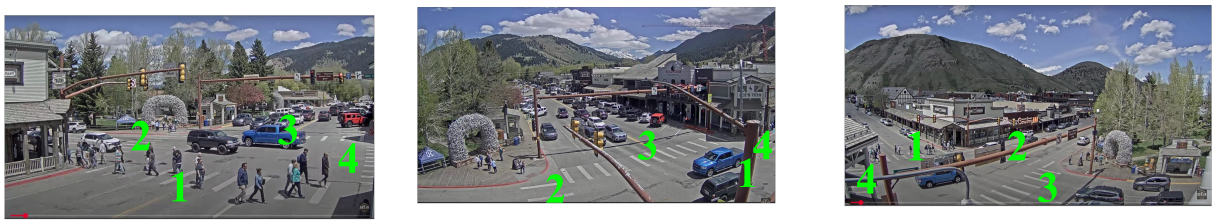

Figure 3: *Task3.*

videos are *synchronised* in the sense that the first frame (with index 0) in both videos corresponds to the same real-world scene in time. Notice that the overall lengths of the two videos can differ. A tight bounding box around the target vehicle is provided in the first frame of camera A, formatted as `[xmin ymin xmax ymax]`, where `(xmin , ymin )` is the top-left corner and `(xmax , ymax )` is the bottom-right corner. Your task is to track this vehicle from the initial frame to the last frame of each video, providing a track in *both* videos (see Fig. 2).

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 30%. The format that you need to follow is the one used in the ground-truth files (located in the sub-directory *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 Task 2. In the test scenario we will release 15 testing pairs of videos. By correctly solving the Task 2 on all videos you will get 1.5 points. Each pair is worth 0.10 points in total: 0.05 points for a correct track in Camera A and 0.05 points for a correct track in Camera B. A correct track in a video is a track in which your algorithm correctly tracks the vehicle.

- **Task 3** – *Directional Traffic Counting*. The folder contains 15 training pairs of *synchronised* videos. In each pair we annotate four canonical approach directions, labelled 1 – 4 in Figure 3. At test time you will be given a single *oriented direction*

$$\text{od} \in \{1\to2,\ 1\to3,\ 1\to4,\ 2\to1,\ 2\to4,\ 3\to1,\ 3\to2,\ 4\to2,\ 4\to3\},$$

defined by its origin direction and destination direction. Your algorithm must return the total number of vehicles that follow this trajectory during the entire video pair.

The format is the following: oriented direction *od* on the first line, number of vehicles on the second line.

Grading scheme for Task 3. In the test scenario we will release 15 testing pairs of videos, each accompanied by one query oriented direction *od*. By correctly solving the Task 3 on all videos you will get 1.5 points. You get 0.10 points/video if your algorithm correctly predicts exactly the number of vehicles in the specified oriented direction. Notice that the overall lengths of the two videos can differ. You need to count every vehicle that completes the specified trajectory *od* in either of the two videos, even if a particular vehicle is visible in only one of them (due to the other video being much shorter in length).

- 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.

The oral presentation of this project (face-to-face or online) will be scheduled after the deadline, in the period $25th - 27th$ of June. It will take around 10-15 minutes in which Alexandra or Bogdan will ask questions regarding implementation. The oral presentation will count for 0.5 points and is mandatory for each student submitting his solution for this project.

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 (Python files or Jupyter notebook files), all auxiliary data that you are using (templates, models, etc.) and a pdf file describing your approach until Tuesday, 24^{th} of June, 11.59 PM using the following link https://tinyurl.com/CV-2025-PROJECT2-SUBMISSIONS. Please do not include in your zip archive any unuseful data (like training videos, we already have them!!!). 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 and where to look for the output file. Students who do not describe their approach (using a pdf file) will incur a penalty of 0.5 points.

On Wednesday 25^{th} of June we will make available the test data. You will have to run your solution on the test videos provided by us and upload your results in the same day as a zip archive using the following link https://tinyurl.com/CV-2025-PROJECT2-RESULTS.

IMPORTANT NOTE. After the deadline, you are not allowed to make any changes that improve your solution on the test data. This includes modifying code parameters, updating trained models, altering templates, or similar adjustments. Please ensure that you test your solution in a newly created environment following your README instructions to confirm that it runs correctly on our machines. We will execute your code to verify that your reported results match our outputs. You may only change non-critical settings, such as adjusting paths for test data or modifying try/except blocks if your code produces an error when processing a test image.