VEHICLE COUNTING IN REGION OF INTEREST THROUGH TRAFFIC SURVEILLANCE CAMERA VIDEO FOR HCMC ROADWAY

Final Project Report - CS117.O21.KHTN

Instructor: PhD. Ngo Duc Thanh

Group 3

Hoang Ngoc Quan - 22521178

Le Trong Dai Truong - 22521576

Nguyen Nhat Minh - 21521135

July 14, 2024

Contents

Chapter 1

Problem Analysis

1.1 Introduction

Over the last few years, traffic congestion has consistently emerged as a significant concern in major urban areas, including Ho Chi Minh City. This pressing concern underscores the critical need for accurate vehicle counting systems to inform effective traffic management strategies and urban planning initiatives. However, current practices for tallying vehicles on HCMC roadways predominantly rely on manual human observation or the installation of stationary sensors, posing obstacles to cost-effectiveness, operational efficiency, and comprehensive information provision.

To address these issues, leveraging video footage obtained from traffic surveillance cameras can be considered as a promising solution. By harnessing this visual data as input, the system can estimate vehicle traffic and provide detailed information about them as output. Moreover, each surveillance camera can be configured with a designated Region of Interest (ROI) to narrow down the area where vehicle counting occurs. The progress in Computer Vision technology and video analytics offers a viable avenue for vehicle traffic counting through traffic surveillance camera video, which enables obviating manual labor, streamlining time consumption, ensuring precision, and delivering cost-effective outcomes.

1.2 Problem Statement

To address the *challenges mentioned above* using the approach of leveraging traffic surveillance cameras, it is imperative to reframe the problem with specific constraints and detailed expectations.

Input:

- A training dataset $\mathcal{D} = \{(image_i, \{bb_j, cls_j\}_i)\}_{i=0}^{N}$, where $cls_j \in \mathcal{L} = \{l_1, ..., l_M\}$, representing annotations for each image i.
- Additionally, a user-defined list of interested vehicle types $\mathcal{T}$ is specified, where $\mathcal{T} \subset \mathcal{L}$.
- The objective is to process a video X to detect and tally the occurrences of the specified vehicle types within a defined region of interest (ROI) delimited by a user-determined polygon $\mathcal{P}$ with $\mathcal{V}$ vertices.

Output:

- A detailed information file encompassing the count of each vehicle type within the ROI from $\mathcal{T}$, along with bounding boxes, object classes, and IDs of objects in each frame of video X that fall within the specified ROI.

1.3 Constraints

To clarify the scope of solving the current problem, we abstract it into several constraints as follows:

Camera:

- The camera's position and viewing angle are fixed, ensuring unobstructed vision to capture clear images. The camera must be capable of producing clear images both day and night, with a high-definition resolution of 720 x 1080 pixels, and the ability to record video at a maximum speed of 60 frames per second (FPS). Regular lens cleaning is essential to prevent the accumulation of dust and dirt.
- Whenever possible, existing traffic surveillance cameras should be utilized to reduce hardware and operating costs.

Vehicle Counting:

- Vehicle types are use-defined and are limited to common transports in Vietnam. The region of interest (ROI) is delineated by a user-defined polygon, specifying the area for vehicle counting.
- Each vehicle that passes through the ROI once and disappears is counted as one count. Vehicles are not obscured completely from each other in the ROI.

1.4 Requirements

The solution aims to achieve positive outcomes in several key aspects:

Processing Speed:

- It must exhibit real-time or near real-time video processing capabilities, operating at a minimum speed of 10 FPS. The minimum hardware specifications require 8 GB of RAM, 2.0 GHz CPU, and 2 GB GPU.

Reliability:

- It is paramount, to mandate precise identification of vehicles of varying sizes, shapes, and orientations within the ROI, with an overall detection accuracy surpassing 90% on the testing dataset.

- Furthermore, the system must maintain a *low Miss Counting Rate*, remaining below 5% under optimal conditions (good lighting and no obstructions and occlusion), below 10% under moderately complex conditions (light rain, thin fog/dust, and no obstructions and occlusion), and below 20% under highly complex conditions (heavy rain, dense fog/dust, minor obstructions, and occlusion) on the testing dataset.

Chapter 2

Solution Evaluation

To ensure that the solution we construct is acceptable and aligns with the constraints while meeting the expectations of the problem, we have opted to select the following dataset and key metrics for quantitative evaluation.

Requirement	**Metric**
Processing speed	FPS (Frame per Second)
Reliability	mAP (mean Average Precision)
Miss Counting Rate reduction	mMCR (mean Miss Counting Rate)

Table 2.1: Requirements and metrics

2.1 Metrics

Frame per Second (FPS):

FPS represents the frequency of frames per second in a video or the rate at which a system can process frames per second.

mean Average Precision (mAP):

mAP computes the mean AP value across all classes. AP computes the Average Precision value for Recall values over 0 to 1 of each class. It is a popular metric to evaluate performance in Object Detection tasks. Steps to compute:

1. Determine the IoU threshold to decide whether the predicted bounding box (bbox) is TP or FP. The predicted bounding box has IoU(predicted bbox, ground truth bbox) $\geq$ IoU threshold is TP, and vice versa.
2. Sort predicted bbox in descending order of their confidence scores.
3. Compute Precision and Recall at each level of the sorted bboxes list.
4. Draw Precision-Recall Curve based on calculated Precision and Recall values.
5. AP is the area under the Precision-Recall Curve, calculate this value by methods: 11-point Interpolation, Area Under Curve – AUC, Interpolated AP.

Intersection over Union (IoU)

IoU is a metric commonly used in object detection to measure the degree of overlap between a predicted bounding box and a ground truth bounding box.

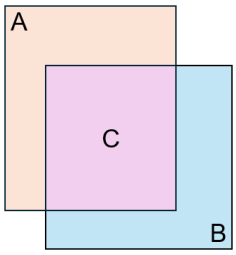

Figure 2.1: Illustration image for Eq. 2.1

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|C|}{|A| + |B| - |C|} \tag{2.1}$$

With A and B are 2 bounding boxes that need to be calculated IoU together, C is the intersection area of A and B.

Precision

Precision at the k-th bbox in the sorted bboxes list is calculated as follows:

$$Precision@k = \frac{\text{num. of TP bbox in the first top k bboxes}}{\text{k}} \tag{2.2}$$

Recall

Recall at the k-th bbox in the sorted bboxes list is calculated as follows:

$$Recall@k = \frac{\text{num. of TP bbox in the first top k bboxes}}{\text{Total num. of ground truth bboxes of the current class}} \tag{2.3}$$

mean Miss Counting Rate (mMCR):

MCR signifies the ratio of actual objects missed by the model to the total actual objects. The mMCR denotes the average MCR of all classes, with MCR computed using the following formula:

$$MCR = \frac{|\text{num. of actual objects} - \text{num. of detected objects}|}{\text{num. of actual objects}} \tag{2.4}$$

2.2 Dataset

The dataset we use for both training and testing is AI Challenge Ho Chi Minh City 2020. It is suitable for this problem because:

- This dataset is acquired from various traffic surveillance cameras positioned overhead, observing the designated roadways within HCM City.
- It encompasses a diverse range of scenarios encompassing distinct vehicle types, traffic densities, and weather conditions.

Chapter 3

Hierarchical Structure

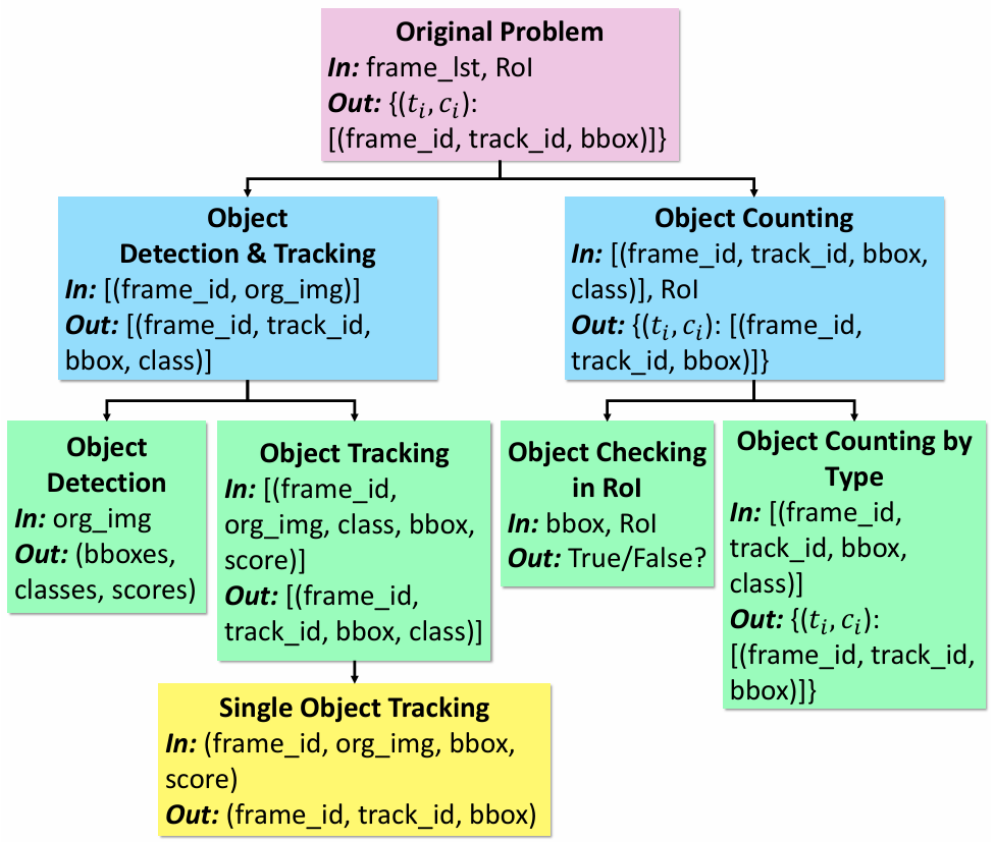

Figure 3.1: Decomposition tree for our problem

The video can be conceptualized as a representation of a sequence of frames, where each frame serves as an encapsulation image of a set of objects within it. Thus, the crucial question lies in how we can effectively recognize these objects, identify their relevance to the list of interested vehicles, and verify their presence within the ROI. By addressing these aspects proficiently, the aforementioned challenge can be elegantly resolved straightforwardly.

The key to addressing these questions lies in the imperative tasks that we need to detect and determine information about objects within the image such as the class and location of them. It is not challenging to immediately associate this with the Object Detection problem, which involves taking an image as input and producing as output the identification of the objects within the image, their respective classes, the bounding boxes encapsulating the objects to denote their locations, and the scores indicating the model's confidence levels.

Following this insight, our crucial aim is to use this data to validate any object belonging to the list of interested vehicles and its presence within the ROI. To be able to do this,

we abstract objects as rectangles created by their bounding boxes. Therefore, we only verify the intersection between the ROI and the object's bounding box to do this.

However, we can not utilize this information to estimate traffic vehicles, because they will always move through frames, and counting objects on each frame can cause duplication. To overcome these issues, we need to perform tracking for objects through frames, and ID assignment for them. It is easy to match this with the problem of Object Tracking, which provides a list of frames of image and class, a bounding box, and a score of the object with the image as input, and receives a track ID of the object as output to maintain continuity in object monitoring.

Finally, filtering out the types of interested vehicles and the list of entities within the ROI, the track ID is leveraged to count vehicle traffic by type based on the classification of the objects. In conclusion, the original problem can be decomposed into the following independent core problems: Object Detection, Object Tracking, Object Checking in ROI, and Object Counting by Type, shown in *Figure* 3.1.

3.1 Object Detection

Given the diverse contextual inputs such as changing positions, varying sizes, and shapes of objects, and different environmental conditions, it is challenging to devise a formal programming approach for this issue. This is where we contemplate Deep Learning strategies, where models are trained on a known dataset to extract features and patterns for generalization and prediction of new input data. There are various model templates we can consider for this problem, but we need to meet the criteria for real-time processing speed and robustness in the road traffic context. Therefore, YOLOv5 might be a suitable solution for this issue.

YOLO (You Only Look Once) [2], introduced in 2015, revolutionized object detection with its single-stage architecture. While earlier methods performed separate object localization and classification steps, YOLO treats object detection as a single regression problem, in which an input image is passed through a CNN to extract features from the image, and then a series of fully connected layers will predict class probabilities, bounding box coordinates and confidence scores for objects within the image. This streamlined approach makes YOLO incredibly fast, and particularly well-suited for real-time applications like traffic analysis.

3.2 Object Tracking

Following the object tracking and ID assignment task, it is innovative to come over the DeepSORT [1] pattern. DeepSORT (Simple Online and Realtime Tracking) is an algorithm used for object tracking in video streams and is an extension of the SORT algorithm. While

the previous version uses the Kalman filter for object tracking, DeepSORT incorporates a deep association metric based on appearance features learned by a CNN. This is the main reason we decided to use this advancement, as it can handle situations where objects may temporarily disappear or become occluded in the video stream.

3.3 Object Checking in ROI

While Object Detection and Object Tracking require the use of Deep Learning strategies and a large amount of data to solve, with the abstraction approach we have presented, it is indeed possible to utilize simple geometric theorems to address the Object Checking in ROI problem. Verifying whether there is an intersection between an object and the ROI can be reduced to the task of checking whether one of the four vertices of the bounding box rectangle lies inside the ROI polygon or not.

For a convex polygon, determining if a point lies within that polygon is simply a matter of calculating the sum of the areas of triangles formed by that point and the polygon's edges compared to the area of the polygon. By applying this abstraction, we can solve this problem elegantly, accurately, and efficiently.

Chapter 4

Algorithm

In summary, all of the analyses mentioned above serve as catalysts driving our solution of a natural step-to-step. The algorithm can be shown by a data active flow diagram (DAFD) in *Figure* 4.1. In the first step, the input video and ROI are organized into a frame list of images and polygon coordinates. By utilizing YOLOv5, we can extract objects within each image frame, along with their location, class, and confidence score. Subsequently, we duplicate each frame based on the number of objects contained within that image, treating each image as if it contains only one object. This data representation process is done to prepare for tracking objects across a sequence of frames. Next, DeepSORT is employed to maintain the tracking of objects and assign IDs to them. Finally, using these ID tags and the extracted object location, we check for their presence within the ROI and count vehicle traffic based on the class of the vehicles of interest. For each interested vehicle type, denoted as t, we record the output of the number of vehicles appearing within the ROI as c, along with a detailed list of the frames where they exist, the assigned IDs, and the location information of where they appear in each frame.

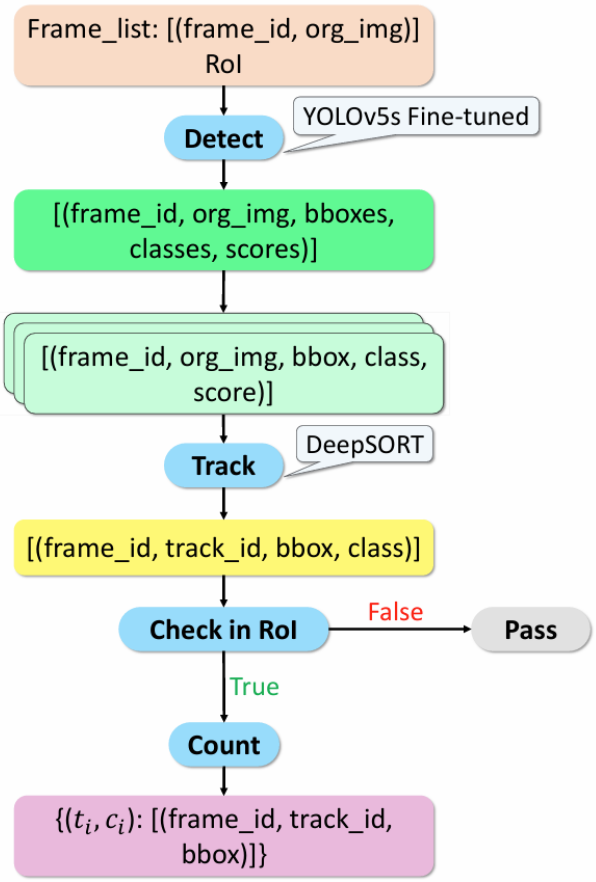

Figure 4.1: Algorithm diagram for the proposed solution

Embarking on the path to achieve this algorithm from the given challenge is like listening to the sound of resilience. Just like different pieces of music come together to create a harmonious symphony, the Decomposition, Pattern Recognition, and Abstraction work in unison in each iteration. This process will be visually presented in the next chapter through a Graphic Organizer.

Chapter 5

Computational Thinking Application

Table 5.1: Iteration 1

Problem Identification	<Context> Traffic congestion has become a significant issue in major cities like HCMC. This highlights the importance of precise vehicle counting systems for better traffic management and urban planning. However, current methods of tallying vehicles in HCMC mainly involve manual human observation or stationary sensors, which can be costly, inefficient, and may not provide comprehensive data.
Pattern Recognition	- The utilization of video footage from traffic surveillance cameras presents a promising solution.
Abstraction	- This system can effectively estimate vehicle traffic by analyzing visual data as input and generating detailed information as output. - Each surveillance camera can be configured with a designated region of interest (ROI) to focus on specific areas for vehicle counting.

Table 5.2: Iteration 2

Problem Identification	What do we need to address the challenge of leveraging the approach of *harnessing traffic surveillance cameras*?
Decomposition	- Collecting video data about traffic roads in HCMC. - Categorizing vehicle types of interest. - Setting a user-defined polygon to establish the ROI in each video.
Pattern Recognition	- The advancement in Computer Vision and Video Analytics technologies provides a feasible approach for vehicle traffic counting in ROI through traffic surveillance camera video.
Abstraction	- Providing a detailed output file with counts of each vehicle type within the ROI, as well as information about their appearance. By abstracting the essential information, comprehensive and structured data enables valuable analysis and decision-making.

Table 5.3: Iteration 3

Problem Identification	What key considerations must be ensured when addressing the challenge *using Computer Vision and Video Analysis technology*?
Decomposition	- **Camera Setup Specifications**: fixed position and angle, clarity in day and night, high-resolution imagery, and a minimum FPS. - **Maintenance Considerations**: regular lens cleaning to ensure optimal performance and prevent image degradation over time. - **Utilization of Existing Infrastructure**: leverage existing traffic surveillance cameras to reduce operation cost.
Abstraction	- **Vehicle Counting Logic**: treat each passing vehicle in the ROI as an independent count, and vehicles are not obscured completely from each other. - **Simplified User Definition**: Abstract user definition to specify vehicle types and delineate the ROI to reduce tasks for the system. Abstract the ROI to a polygon and limit vehicle types to common road transports in Vietnam.

Table 5.4: Iteration 4

Problem Identification	How to *evaluate the effectiveness and suitability* of the solution when addressing the challenge?
Decomposition	- **Processing Speed**: real-time or near real-time video processing. - **Reliability Assessment**: focusing on precise vehicle identification within the ROI and reducing miss-counting rates under different environmental conditions for effective system performance. - **Scalability**: overcome diverse contextual inputs such as vehicles of varying sizes, shapes, and orientations, and different conditions.
Pattern Recognition	- Using some template metrics for quantitative evaluation: **FPS (Frame per Second)**: reflects the rate at which the system can process frames per second. **mAP (mean Average Precision)**: reflects the overall detection accuracy. **mMCR (mean Miss Counting Rate)**: signifies the ratio of actual objects missed to count by the system. - Collecting a dataset encompasses a diverse range of scenarios, specifically, it is acquired from HCMC roadways.

Table 5.5: Iteration 5

Problem Identification	<Main> How to solve the problem of vehicle traffic counting in the region of interest (ROI) through traffic surveillance camera video using Computer Vision and Video Analysis techniques acceptable and aligns with the constraints while meeting the expectations mentioned above?
Abstraction	- The video can be conceptualized as a representation of a sequence of frames, where each frame serves as an encapsulation image of a set of objects within it.
Decomposition	- How can we effectively recognize objects, and identify their relevance to the list of interested vehicles? - How can we verify the object's presence within the ROI? - How can we avoid duplicates when counting?

Table 5.6: Iteration 6

Problem Identification	Vehicles always move through frames, and counting objects on each frame separately can cause duplication.
Pattern Recognition	- Perform tracking for objects through frames, and ID assignment for them. The track ID is leveraged for estimating vehicle traffic to avoid repetition when counting.

Table 5.7: Iteration 7

Problem Identification	<Sub1> Recognize objects, and identify their relevance to the list of interested vehicles: need to detect and determine information about objects within the image such as their class and location. Pay attention to meeting the criteria for various scalability of the input, real-time processing speed, and robustness in the road traffic context.
Pattern Recognition	- **Object Detection**: taking an image as input and producing as output the identification of the objects within the image, their respective classes, and locations. - **YOLOv5** [2]: a single-stage architecture, it treats this problem as a regression problem. Thus, YOLO is incredibly fast and particularly well-suited for real-time applications like traffic analysis.
Abstract	- Abstracting objects from the 3D real world to the 2D image as rectangles created by their bounding boxes. - The likelihood of prediction reliability is represented by a confidence score of the model.

Table 5.8: Iteration 8

Problem Identification	<Sub2> Verify the object's presence within the ROI.
Pattern Recognition	- It is indeed possible to utilize simple geometric theorems. - Calculating the sum of the areas of triangles formed by that point and the polygon's edges compared to the area of the polygon.
Abstract	- Verifying the intersection between the ROI and the object's bounding box. - Checking whether one of the four vertices of the bounding box rectangle lies inside the ROI polygon or not.

Table 5.9: Iteration 9

Problem Identification	<Sub3> Perform tracking for objects through frames, and ID assignment for them. Besides, objects may *temporarily disappear* or become *occluded* in some frames in the video stream.
Pattern Recognition	- **Object Tracking**: provides a list of frames of image and class, bounding box, and score of the object with the image as input, and receives a tracking ID of the object as output. - **DeepSORT** [1]: is an extension of the SORT algorithm, incorporates a deep association metric based on appearance features learned by a CNN to improve the robustness of the *disappearance* and *occlusion* situation.
Abstract	- To meet with the multi-class of object tracking, create several trackers corresponding to the number of interested vehicle types and track them according to their class.

Chapter 6

Discussion and Conclusion

6.1 Visualization

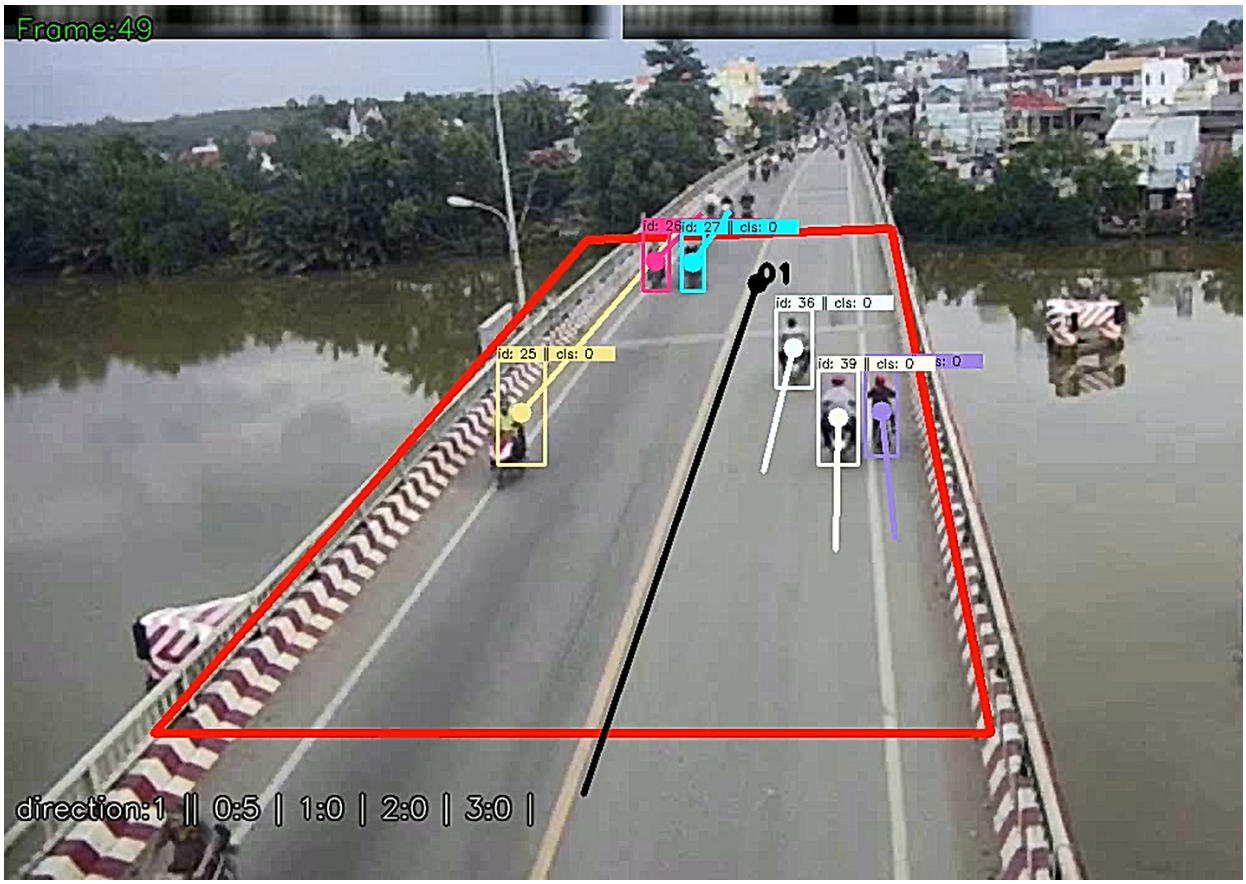

Figure 6.1: Visualize how our solution works

We utilize existing techniques to visualize *optionally* the operation of our solution on a video obtained from an available traffic surveillance camera. The information displayed in the figure is derived from the output result file, shown in *Figure* 6.1. The red-bordered polygon indicates the Region of Interest (ROI) predefined by the user. All traffic objects within the ROI are localized (represented by bounding boxes), classified, and assigned an ID during tracking (represented by ID and cls corresponding to each bounding box). The count of each type of interested vehicle passing through the ROI is displayed in the bottom left corner. The information shown includes the index of the kind in the list of vehicles of interest and the corresponding count of that vehicle type that has passed through the ROI.

6.2 Discussion

Manual methods are costly in terms of time and labor, often leading to errors. On the other hand, sensor-based approaches require significant investment in technologies like IoT and regular hardware maintenance to ensure the best accuracy. In contrast, a Computer Vision-based approach using traffic monitoring video cameras offers a balance between efficiency, accuracy, and operational costs. In particular, this method provides valuable vision evidence by identifying moving vehicles within the ROI, including their locations, timings, types, and rather than only results.

Another approach using Computer Vision to count vehicles and prevent duplication through Object Detection focuses on counting objects only when they intersect with the ROI edges, regardless of their position inside or outside the ROI. While this simplifies Object Tracking compared to the proposed solution, it does require more video frames to capture objects precisely when they touch the ROI boundary, thus increasing processing time. Additionally, ROI boundaries often coincide with traffic junctions, leading to potential issues such as bottlenecks, obstructions, and occlusion to detect accurately. Besides, utilizing Computer Vision technologies to automatically define the ROI, rather than relying on user-defined parameters, could increase system complexity. This approach may need an additional preprocessing module for ROI identification in videos. Even so, it could be a potential area for our future work.

6.3 Conclusion

By leveraging computational thinking skills such as Decomposition, Pattern Recognition, and Abstraction and then formatting them as an algorithm, we have successfully devised a viable solution to the challenge. Through practical experiments, the model can run in real-time on a device hardware configuration: 8GB RAM, 2.0GHz CPU, 2GB VRAM GPU (NVIDIA MX250). However, the issue of its reliability in use still needs to be tested and evaluated further, before the method is applied in practice.

In this final project, we worked together to seek challenges from reality, analyze them into a problem clearly, and explore various approaches in their advantages and disadvantages. Based on these analyses, we decomposed the main problem into subproblems, abstracted them, and integrated them with existing patterns. It is a wonderful time to exchange ideas and gain a deeper understanding of applying Computational Thinking principles such as Challenge Finding, Problem Analysis, Decomposition, Pattern Recognition, Abstraction, Solution Evaluation, and Algorithm Building. As a result, we feel grateful for this CS117 course. The entire source code and project demo are published on this Github link.

References

[1] Alex Bewley et al. “Simple online and realtime tracking”. In: *2016 IEEE international conference on image processing (ICIP)*. IEEE. 2016, pp. 3464–3468.

[2] Joseph Redmon et al. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.