Video Surveillance for Road Traffic Monitoring

Caballero, Ana ana.caballeroc@e-campus.uab.cat
Vallve, Arnau arnau.vallve@e-campus.uab.cat
Malak, Marcin malak.marcin@e-campus.uab.cat
Jaworski, Wiktor jaworski.wiktor@e-campus.uab.cat

***Abstract*—This work proposes a framework to learn the basic concepts and techniques related to video sequences mainly for surveillance applications and can be applied to any problem in order to obtain accurate automatic results. The system is based on the NVIDIA AI CITY Challenge [1]. We started with a foreground estimation, then post processing techniques are applied to the detect and track vehicles in motion. Finally, we stabilized video with Optical Flow and applied all we learned to multiple cameras. The project has been developed in Python [2]. However, this report is focused just in the last week of the work.**

I. Introduction

THE use of intelligent traffic surveillance systems, based on algorithms and computer vision systems are very useful at present to improve the problems of traffic congestion and any anomaly that may occur on the roads such as accidents or dangerous infractions. These systems, based on numerical data, process the information received through the network of video cameras already used for a long time to check traffic on the roads and extract information in real time for operators to process and collect statistics to improve traffic flows. traffic.

These systems are essential in our times due to the great increase in traffic and the human difficulty of processing so much information in real time and giving an immediate response to solve the problem.

II. The NVIDIA AI CITY CHALLENGE

The aim of this challenge is to encourage the research and development techniques that rely less on supervised approaches and more on transfer learning, unsupervised and semi-supervised approaches that go beyond bounding boxes. It is focused on **Intelligent Transportation System (ITS)** to satisfy the need of making transportation systems smarter.

Nvidia proposes three challenges with different datasets provided as a set of videos or images, as images in figure 1. These challenges are:

- City-scale multi-camera vehicle tracking
- City-scale multi-camera vehicle re-identification
- Traffic anomaly detection

Out project is focused on the first track challenge: **City-scale multi-camera vehicle tracking**.

Under the guidance of Javier Ruiz and Xavier Giro.

(a) Image frame example 1 (b) Image frame example 2

Fig. 1: Example images from Nvidia AI City Challenge web

III. Multi-Target Single-Camera Tracking (MTSC)

There are various solution which allow us to implement multi-target single-camera tracking. For the needs of the subject we tried to implement one of provided and suggested solution:

- **Tracking by overlap**
- **Kalman filtering**

For the first idea, we have been using the concept of adding for each detected car new ID. We have already created for that two different model of BoundingBox and Frame which store data necessary to creating detection for provided video(a). After implementing everything, we figure it out that using very big IoU threshold (0.95) is not working properly(b), because detections with IoU ¡ 0.95 are seen and labeled as new detections. Non in-depth qualitative tests suggest the IoU threshold values close to 0.5 give the best results. High values of look-back frames give bad results for dynamic objects and way better results for non-moving objects. The two parameters could be tweaked to bring the best results for given scenario.

(a) Overlap example 1 (b) Overlap example 2

As a second approach we tried to implement Kalman filtering. We end up in the situation where prediction is slightly good, but in some particular cases there are mistakes of prediction. We are considering that this problem might occur

because of the linear nature of the filter. There are different type of movement of cars, not always in linear way.

We also had some difficulties to find out the reason why our filter in some part of detection changes bbox id for other one (which we think it proves that filter is missing detection)

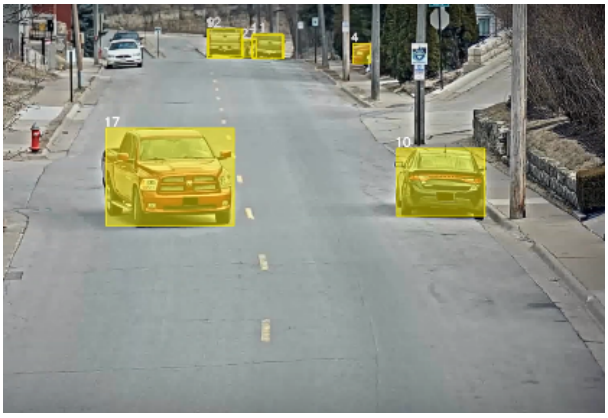

Fig. 3: Kalman filtering

Both of the situation described above occur mostly with the strange and not sufficient results. With all the metrics we have been trying to achieve, we get a strange results. Very different results than expected.

The table below show up the results of metrics we achieved:

	IDF1 (SEQ 3)						
Camera	c10	c11	c012	c013	c014	c015	Average
Kalman	0,289	0,44	0,758	0,566	0,378	0,33	**0,46**
IOU	1,237	1,867	2,479	2,127	1,974	1,469	**1,85**

Fig. 4: Results table

IV. MTSC AND TRACKING BY OVERLAP

Multiple target single camera tracking is the city environment is a very challenging task. It is because of dynamic environment and many possible occlusions that often lead to many errors such as swapping IDs of the objects when occluding. In our case we tried to keep track of each object using **Kalman filter** or *Overlap tracking*. As seen in Figure 7 the performance of the the methods was drastically different. Kalman tracking uses the propablisitic methods to assign the same object ID between various frames (from the point of view of Kalman filter - states) to each detected object. For each detection the ID is assigned based on the given error covariance of the prediction. The Overlap simply calcualtes the intersection over union metric between each detection of two frames. The ID is propagated from previous frame to next frame based on the higher IoU score. In our implementation of tracking using IoU we experienced problems when new objects were appearing and that led to bad results. We also tested qualitatively the threshold values (that is minimum IoU to assign the existing ID to an object) and we obtained the best results for values near *0.4*. Kalman provided results more stable but still we experienced many re-detections - that is perceiving the occluded object in the consecutive frames as new detection and assigning new ID to it.

V. MULTI-TARGET MULTI-CAMERA TRACKING (MCMT)

Multi target multi camera tracking was the final goal of the projects. To achieve it we need to:

- Detect objects - that is done by fine-tuned off-the shelf object detection system model **yolov3**
- Track targets - done as mentioned in chapter section IV
- Re-identify targets, using methods mentioned in chapter section VI or using existing systems for car re-identification such as veri-776
- Assign the same ID to targets across multiple cameras.

VI. MCMT ROBUST HOUGH-BASED HOMOGRAPY PROJECTIONS

One of the first approaches we tried in order to work on the multi-camera domain was based on the paper Multi-camera Multi-object Tracking by Robust Hough-based Homography Projections [3] for which no implementation was found, so we tried to implement it by ourselves. The method uses a generalized Hough voting and extends it into a multi-camera domain which is what we require for our problem. This method has the advantage that requires a low amount of data for the voting procedure, it is robust to projection errors and utilizes geometric information to improve the performance of the tracking.

The method revolves around the use of the Hough transform. As the Hough transform allows to find certain shapes on an image using a voting procedure it seems suitable for the task of finding object candidates. However, this method uses a new voting scheme for which the voting results are fused over the multiple cameras. Additionally it also uses geometric verification and backprojection between views to improve the tracking and applies a particle filtering approach to avoid overlapping particles.

The method can be visually understood by looking at figure 5. While most methods based on background subtraction can have projection errors and ghosting detections, the Hough multi-camera voting approach generates Hough maps from voting in each of the cameras we have, which are then projected into a common ground plane where the geometric uncertainties are implicitly considered.

The blocks required in order to make it work are the following. First, a detector must be used in order to obtain the objects we need to track. From there it is required to map the obtained votes from each of the camera views on the common ground plane using the homographies he obtain between the camera and the ground plane. For the voting Hough Forest learning should be used, and the Hough maps should consider voting foot-points instead of the typically used centroids. For the tracking, we want to use multi-object particle filtering with the prior that we don't want objects to occlude in the general plane.

After getting to understand the intricacies of the method and starting to see the issues arising when implementing it

from scratch, due to time constraints, it was not possible to evaluate it with our data and implement it for our project. Several alternatives were then also considered.

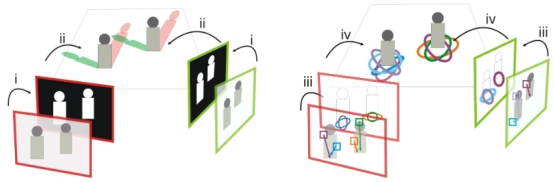

Fig. 5: Approach based on background subtraction (left) vs Hough multi-camera by joint votes [3] (right)

VII. FEATURES FOR MCMT AND RE-IDENTIFICATION

DukeMTMC (Duke Multi-Target, Multi-Camera Tracking Project) aims to accelerate advances in multi-target multi-camera tracking. It provides a tracking system that works within and across different cameras, as in figure6, and a new performance evaluation method that measures how often a system is correct about who is where.

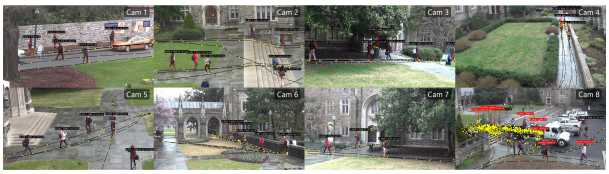

Fig. 6: DukeMTMC is a new, manually annotated, calibrated, multi-camera data set recorded outdoors on the Duke University campus [4]

Features for Multi-Target Multi-Camera Tracking and Re-Identification [5], is a paper developed by the a group of researchers from Duke University in United States about tracking many people through video taken from several cameras.

Their algorithm learns features using an adaptive weighted triplet and assigns adaptive weights using the soft-max/min distribution. Finally an identity label is assigned to the detected observations (people). The model is trained using an ImageNet network pretrained using data augmentation. Trajectories are computed online in a sliding window, finally the ones with low confidence are removed.

The authors run different experiments, including:

- Measure overall MTMCT performance.
- Measure the impact of improved detector and features during tracking.
- Study the relation between measures of accuracy.
- Demonstrate the usefulness of the methods used.
- Analyze tracker failures.

We tried to understand the paper and apply the same solution to cars but we didn't succeed because some details were difficult to understand. In addition, we don't know the trajectories in advance and this is an input needed. But, what made us give up with this paper was the fact that training the network, as they explain in the paper, was very cost expensive and we were not able to make Google Cloud [6] work in previous weeks.

VIII. MCMT II WORKSHOP AND CHALLENGE

Regarding the previous section, we found a code in Matlab uploaded in a github repository [7]. The code follows the steps described in section VII. It is the code of the PhD disertation from Ergys Ristani who also participates in the motchallenge [8] providing technical support. This challenge pretends to pave the way for a unified framework towards more meaningful quantification of multi-target tracking. This objective is similar to the Nvidia Challange, but they focus in people instead of vehicles.
We spent a couple of hours trying to understand what he does and testing with our video but the results were awful with the pretrained weights.

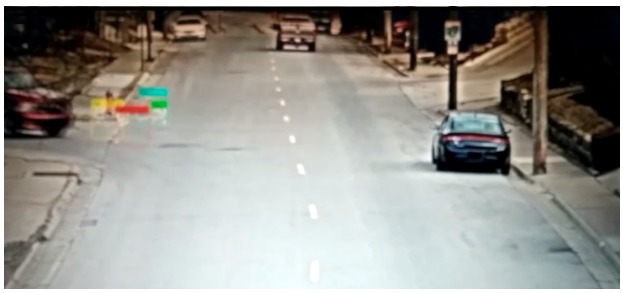

Fig. 7: II Workshop and Challenge, code tested over our video. False detections are observed at the left of the images

Finally we discarded this algorithm because to understand it completely download 160 GB of data was needed.

REFERENCES

[1] Nvidia, "Nvidia AI City Challenge," .
[2] 2019 M6 Team 8, "M6 Video Analysis. Git Repostory," .
[3] Sabine Sternig, Thomas Mauthner, Arnold Irschara, Peter M Roth, and Horst Bischof, "Multi-camera multi-object tracking by robust hough-based homography projections," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE, 2011, pp. 1689–1696.
[4] DukeMTMC, "DukeMTMC Data Set," .
[5] Tomasi Carlo Ristani, Ergys, "Features for multi-target multi-camera tracking and re-identification," *Computer Vision Foundation - open access*, Dec. 2018.
[6] Google, "Google Cloud," .
[7] ergysr, "Multi-Target, Multi-Camera Tracking," .
[8] MOTChallenge: The Multiple Object Tracking Benchmark, "DukeMTMC Data Set," .