AI City Challenge 2020 – Computer Vision for Smart Transportation Applications

Ming-Ching Chang [1] Chen-Kuo Chiang [2] Chun-Ming Tsai [3]
Yun-Kai Chang [2] Hsuan-Lun Chiang [2] Yu-An Wang [2] Shih-Ya Chang [2]
Yun-Lun Li [2] Ming-Shuin Tsai [2] Hung-Yu Tseng [2]

[1] University at Albany – SUNY, NY, USA
[2] National Chung Cheng University, Taiwan
[3] University of Taipei, Taiwan

Abstract

We present methods developed in our participation of the AI City 2020 Challenge (AIC20) and report evaluation results in this contest. With the blooming of AI computer vision techniques, vehicle detection, tracking, identification, and counting all have advanced significantly. However, whether these technologies are ready for real-world smart transportation usage is still a open question. The goal of this work is to apply and integrate state-of-the-art techniques for solving the challenge problems under a standardized setup and evaluation. We participated all 4 AIC20 challenge tracks (T1 to T4). In T1 challenge, we perform vehicle counting by associating deep features extracted from Mask-RCNN detections and tracklets, followed by vehicle movement zone matching. In T2 challenge, we perform vehicle type and color classification and then rank matching vehicles using a PGAM re-id network. In T3 challenge, we proposed a new Multi-Camera Tracking Network (MTCN) that takes single-camera vehicle tracking as input, and performs multi-camera tracklet fusion and linking, by jointly optimizing the matching of vehicle appearance and physical features. In T4 challenge, we adopt a leading method based on perspective detection and spatial-temporal matrix discriminating, and improve it with background modeling for traffic anomaly detection. We achieved top-6 and top-4 performance for T3 and T4 challenges respectively in the AIC20 general leaderboard.

1. Introduction

AI deep neural networks have advanced significantly in recent years, leading to a fast-pacing “smarter world”. Among many advancements, smart city and smart transportation are on the emerging fronts. Intelligent devices with cameras running computer vision (CV) techniques are able to see and start to reason and understand the world. Particularly, under the umbrella of Intelligent Transportation Systems (ITS) developments, the AI City Challenge Workshops [1] are organized with the aim to encourage research and development of AI and CV for smart transportation applications. The AI City Challenge 2020 (AIC20) is the forth sequel following the growing participation from past years (AIC17 [16], AIC18 [17], and 19 [18]), targeting at four challenge tracks in the following:

- Track 1 challenge focuses on the counting of two classes of vehicles (trucks and passenger cars) at multiple intersections observed from various camera views — a dataset provided by the Iowa Department of Transportation (DOT). The key challenge is on how best to reliably track vehicles and determine the crossing of vehicles at specific traffic lanes or zones.
- Track 2 challenge focuses on image-based re-identification of CV detected vehicle boxes from the CityFlow Vehicle Re-Id Dataset [20].
- Track 3 challenge is on the tracking of vehicles over a city-wide camera network that spans over 4 miles from 5 collection sites — the CityFlow dataset [20]. The challenge is on: *(i)* how best to perform multi-camera tracking on synchronizing and overlapping camera views, *(ii)* how best to re-identify vehicle tracks across camera views with large viewing variabilities, and *(iii)* how best to leverage traffic flow characteristics to achieve a reliable solution for the re-identification and linking of vehicle tracklets among the vast number of potential candidates.
- Track 4 challenge is on the detection of abnormal traffic incidences from a dataset provided by Iowa DOT, where anomalies arisen from emergencies, vehicle breakdowns, or crashes.

[1] https://www.aicitychallenge.org/

This paper describes methods and results of our participation submitted to all four AIC20 challenge tracks, with evaluation performed by the AIC20 organization. On the AIC20 leaderboard, our method ranks **18-th** in the general leaderboard (score 0.3241) and **13-th** in the public leaderboard (score 0.3116) in the Track 1 vehicle counting contest. We rank **57-th** in the general leaderboard (**41-th** in the public leaderboard) with mAP 0.0368 in the Track 2 vehicle re-id contest. We rank **6-th** in the general leaderboard out of 9 participant teams (also **6-th** in the public leaderboard out of 8 teams) with score 0.0620 in the Track 3 contest. We rank **4-th** (out of 13 participant teams) with F1-score of 0.9706, RMSE 6.6058, S4 94.92% in the Track 4 anomaly contest in the AIC20 general leaderboard.

T1 Challenge: Multi-Class Multi-Movement Vehicle Counting. Our vehicle class-specific counting pipeline consists of three steps (as in Fig. 1): (1) vehicle type detection using Mask-RCNN [6], (2) appearance and spatial feature-based tracking, and (3) the matching between vehicle Movement of Interest (MOI) zones. The Mask-RCNN vehicle detection can handle large variabilities of image qualities and vehicle scales in the provided videos. Vehicle tracklets are formed based on a standard Hungarian matching of both spatial (geometrical) features and the re-id features following [10]. We train this feature extraction network on both the AIC20 Track 2 and Track 3 challenge datasets, with an extra resizing step to enhance the detection of small-size vehicles appearing in the videos. Finally, vehicle trajectories are matched against specified MOI zones (traffic lanes characterizing the respective vehicle counting tasks) to determine the travelling directions and zone-crossing counts to produce the final counting.

T2 Challenge: City-Scale Multi-Camera Vehicle Re-Identification. Given a query vehicle image and a potentially very large gallery of test images, the vehicle re-id output is the matching vehicles in the gallery ranked by the matching similarities in decreasing order. Our vehicle re-id method consists of two steps (as in Fig. 3): (1) We first performed *supervised vehicle type and color classification*. We annotated the AIC20 T2 vehicle re-id training set for such information. This annotation effort includes the labeling of $15,000$ vehicles that are categorized into **7** vehicle types, **30** vehicle makes, and **9** vehicle colors (see § 3.2 for details). (2) We then trained a *vehicle metadata classifier* based on this dataset to identify vehicle types, makes and colors, which will be used fine-select a subset of gallery vehicles to perform re-identification. Finally, we apply the Pyramid Granularity Attentive Model (PGAM) re-id network [2], a method we developed in the AIC19 contest [18]. The PGAM consists of a BNneck with ResNet-154 backbone that produces the final re-id ranking results.

T3 Challenge: City-Scale Multi-Camera Vehicle Tracking. We performs data-driven multi-cam vehicle tracking in the following steps (see Fig. 6): (1) Vehicles are detected using Mask-RCNN [6] and re-id appearance feature extraction using ResNet-50. (2) Physical measures are obtained using the provided camera calibration matrices to project the vehicle boxes onto a global (longitude, latitude) GPS coordinates. (3) Hungarian matching are performed based on a loss including a *cross-entropy* term using the vehicle re-id appearance feature and GPS positions, and a triplet loss for metric learning. Single camera tracking is performed by bottom-up linking of vehicle tracklets within each camera. (4) Multi-camera tracking are performed using a newly proposed Multi-Camera Tracking Network (MCTN) to associate vehicle tracklets across views, with the optimization of both the physical and appearance feature loss terms.

T4 Challenge: Traffic Anomaly Detection. Anomaly detection in the real-world traffic scene has many challenges, including variations of weather, view-points and lighting conditions, that will affect the accuracy and reliability. Our pipeline is based on the winning method of Bai *et al.* [1] from AIC19 contest [18], with improvements and integration of new capabilities in the vehicle detector and background modeling modules.

2. Related Work

Vehicle detection is important in smart transportation and traffic surveillance. In recent years, methods based on deep learning have improved significantly. Notably, deep Convolutional Neural Network (CNN) based method such as Mask-RCNN [6] and YOLOv3 [19] are widely used in many systems.

Single-camera multi-target tracking methods [4, 8, 5, 15, 14] mostly follow the *tracking-by-detection* paradigm, in associating per-frame object detections into consistent tracklets. Occlusion recovery and tracklet identity switch avoidance are the challenges for these algorithms. Popular tracking methods include *correlation filter* based approaches (KCF [8], SRDCF [5], ECO [4]) and *CNN-based* approaches (DeepSORT [26], MDNet [15], TCNN [14]).

Multi-camera multi-target tracking involves not only Multiple Object Tracking (MOT) but also two additional difficulties/enhancements: (1) How best to associate tracklets across synchronous, overlapping camera views, where camera calibration is often used to reason about the geometry among tracklets in physical world coordinates. (2) How best to link tracklets across non-overlapping camera views, which is essentially the re-identification problem across cameras. Popular methods typically rely on a form of spatial-temporal inference across camera views [25, 10] or ensemble fusion [21].

Vehicle re-identification has drawn significant attentions, as method developed for *e.g.* person re-id can be directly applied to vehicle re-id. However real-world vehicle re-id remains challenging due to various factors: (1) the unbalanced intra- and inter- class variabilities, (2) vehicles of the same make, even same model year, can belong to different owners, where only tiny difference such as the windshield tags is the sole hint to distinguish them, (3) the potentially huge number of vehicles to compare with. Notably, leading methods from AIC19 [9, 13] rely on three general strategies: (1) vehicle keypoint identification (that can lead to 3D modeling of vehicles) that can improve view-invariant feature extraction, (2) leveraging vehicle type, make, and color classification for re-id [9], and (3) *multi-stage re-id*: re-ranking after an initial ranking (often obtained via a triplet-loss re-id network) using extra information or metadata. Popular dataset for training vehicle re-id dataset includes: VRIC (based on UA-DETRAC), PKU VehicleID, and VeRi datasets. [2]

Vehicle counting is traditionally achieved by deploying coil sensors under the road [22]. With the advancement of CV, AI visual systems can now perform non-intrusive vehicle counting based on object detection and tracking [27], which can be deployed at large scales. Challenges of CV vehicle counting include image quality and resolution limitations, view angle variations, occlusions, and weather conditions (raining or snow).

Traffic anomaly detection from real-world traffic videos is not straightforward, where standard data-driven abnormal event detection or outlier detection algorithms (SVM, isolation forests) can easily fail. The winning teams in AIC19 [18] use optical-flow or background modeling, to segment out a moving traffic mask that can effectively reduce search. Abnormal events such as a stalled vehicle can be detected using a spatio-temporal anomaly matrix [1], or a multi-stage framework based on anomaly candidate identification [23].

3. Method

We describe each method developed for the four AIC20 challenge tracks in the following sections.

3.1. (T1) Multi-Class Multi-Movement Vehicle Counting

We perform multi-class multi-movement vehicle counting on the AIC20 Track 1 challenge dataset, which contains annotated videos viewing urban intersection and highway. We adopt a winning method of Li *et al.* [10] from AIC19, which uses Hungarian matching for vehicle track linking. Since the AIC20 Track 1 dataset does not come with camera calibration, we directly use image pixel coordinates (instead of the GPS coordinates in [10]). Our vehicle counting pipeline in Fig. 1 consists of steps that we will describe in the following two parts.

(i) Vehicle detection and tracklet linking (§ 3.1.1). We first use Mask-RCNN to detect vehicles and extract re-id appearance features from each vehicle in order to form tracklets for use in later steps. We use a standard Hungarian matching algorithm to associate detections into tracklets, by considering both the *spatial* and *appearance* features. After such bottom-up vehicle tracklet linking, tracklets can still be broken due to poor detection results, *e.g.* tiny vehicles, occlusions, or camera shaking. To this end, we enforce a tracklet matching and linking step to recover broken tracklets that can hopefully connect into longer trajectories.

(ii) Vehicle movement matching and counting (§ 3.1.2.) Given the refined vehicle tracklets, we next match each vehicle trajectory against the Movement of Interest (MOI) given from the AIC20 contest for each site. The vehicle traveling direction is determined as a classification problem, and the final traffic counting results are obtained.

3.1.1 Vehicle detection and tracklet linking

We adopt the tracking-by-detection method of [10], where Mask-RCNN [6] is first perform to detect vehicles. We confirmed that Mask-RCNN produces more accurate detections compared to other detectors such as YOLOv3[19], especially for small vehicles. Tracklets are next associated based on a fusion of spatial and re-id appearance features. This re-id appearance feature extraction network is based on a Resnet50 backbone with a bottleneck structure [29]. We train this network using the AIC20 Track 2 re-id dataset with both the triple and classification losses.

We found the tiny vehicles (around 10 to 20 pixels) in the view is difficult to detect in the setting of [10]. To this end, we initialize the feature extractor network using pre-trained weights, and fine-tune it using the down-sampled vehicle boxes obtained from the AIC20 Track 2 and Track 3 training sets.

Tracklets are created and linked from detected vehicles between consecutive frames. We calculate the distance (loss) between each vehicle box b_i and its corresponding box b_j in the previous frame. The distance between such pair (b_i, b_j) is calculated as:

$$d(b_i, b_j) = ||p_1 - p_2||_2 + \lambda_p ||f_1 - f_2||_2, \quad (1)$$

where p_1 and p_2 are pixel coordinates; f_1, f_2 are the re-id features in 2048 dimension; λ_p is set as 0.01. Standard Hungarian matching is applied to link these vehicle box pairs by minimizing association distances, which yields off-line vehicle tracking trajectories.

We observed that the cameras can appear to be shaking in the AIC20 Track 1 test set, which hinders Mask-RCNN from producing reliable vehicle detections. To address this issue, we improved the tracklet handling steps in [10]. Specifically, in the case when a tracklet does not

[2] Available at https://github.com/knwng/awesome-vehicle-re-identification.

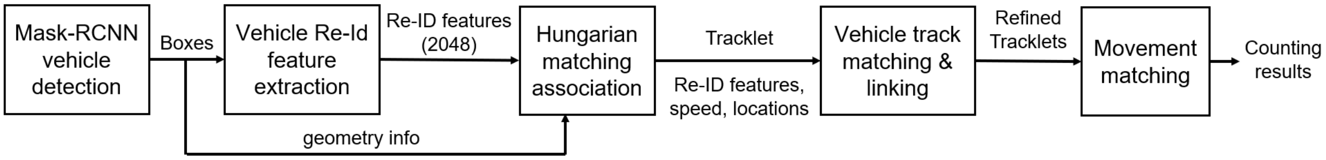
Mask-RCNN vehicle detection
Boxes
Vehicle Re-Id feature extraction
Re-ID features (2048)
Hungarian matching association
Tracklet
Re-ID features, speed, locations
Vehicle track matching & linking
Refined Tracklets
Movement matching
Counting results
geometry info

Figure 1. **Track 1 multi-movement vehicle counting** method overview.

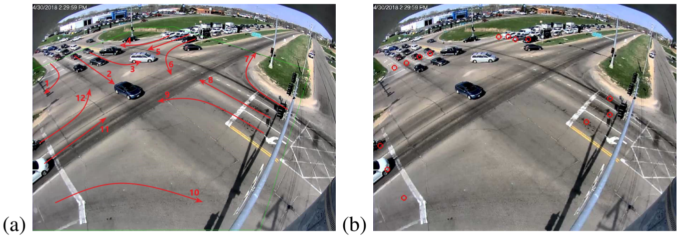

Figure 2. **Track 1 multi-movement vehicle counting** in the specified Movement of Interest (MOI) zones. (a) Each MOI corresponds to a traffic lane, where vehicle counting is to be calculated. (b) We match each vehicle trajectory against each MOI, by calculating the spatial distance of each vehicle box to each MOI starting point.

match to any detection boxes in a frame, we store the tracklet rather than discard it immediately. We discard tracklets in the case where the number of times with no matching detections exceeds a threshold $\tau = 5$.

3.1.2 Vehicle movement matching and counting

We match each vehicle trajectory against a set of Movement of Interest (MOI) zones specified by the AIC20 contest organization as in Fig. 2(a). MOI is used to determine vehicle passing counts as the traffic flows.

To deal with erroneous counts resulting from tracking losses, broken tracks, or ID switches, we enforce tracklet linking to recover likely disconnected tracklets. We consider each tracklet pair (T_a, T_b), where T_a denotes the tracklet appearing earlier than T_b. If the minimum loss distance of last five boxes of T_a and the first five boxes of T_b is lower than a predefined threshold, we re-connect such tracklet pair (T_a, T_b). We also connect (T_a, T_b) if the Euclidean distance between the first box of T_b and the last box of T_a is lower than a threshold τ_p of 400 pixels.

Matching MOT starting points (Fig. 2). To match each vehicle tracklet to a MOI zone, we match the first vehicle position in the tracklet with a manually labeled starting point of MOI. We manually specify such starting point for each traffic lane in each camera view. The distance between each MOI starting point (x_s, y_s) to the center of each vehicle box x_v, y_v is calculated using Manhattan distance in pixels:

$$d_{sv} = |x_s - x_v| + |y_s - y_v|. \quad (2)$$

3.2. (T2) City-Scale Multi-Camera Vehicle Re-Identification

Our vehicle re-id pipeline is a two-stage approach. In the first stage, we propose a vehicle metadata (type, make and color) learning network (§ 3.2.2) that is trained on a newly labeled dataset (§ 3.2.1). Specifically, for the training of the vehicle metadata recognition network, we performed annotation of the vehicle types, makes, and colors on the AIC20 CityFlow Vehicle Re-Id Dataset [20]. We did not make use of the AIC20 VehicleX synthetic dataset. In the second stage, we adopt the Pyramid Granularity Attentive Model (PGAM), an improved re-id network from our previous work of [2] in AIC19 (§ 3.2.3). The pipeline is shown in Fig. 3.

3.2.1 A newly labeled AIC20 vehicle metadata set

We annotate the following 3 attributes on the $15,000$ vehicle images in the AIC training set:

- **Vehicle type:** Sedan, SUV, truck, minivan, pickup truck, hatchback, bus; totally 7 types.
- **Vehicle make:** Dodge, Ford, Chevrolet, GMC, Honda, Chrysler, Jeep, Hyundai, Subaru, Toyota, Buick, KIA, Nissan, Volkswagen, Oldsmobile, BMW, Cadillac, Volvo, Pontiac, Mercury, Lexus, Saturn, Benz, Mazda, Scion, Mini, Lincoln, Audi, Mitsubishi, Others; totally 30 makes.
- **Vehicle color:** black, white, gray, blue, red, gold, silver, green, yellow; totally 9 color types.

Our annotated vehicle dataset will be released upon the acceptance of this paper.

3.2.2 Vehicle type and color metadata learning

We use the 29-layer light CNN framework from Wu *et al.* [9] to perform vehicle metadata classification. In this architecture, Max-Feature-Map (MFM) is used to replace the original ReLU function. MFM can be regarded as a way of maxout, which makes low-activation neurons robust to noise, thus leads to meaningful features. The Semantic Bootstrapping in the architecture can handle noisy labeled images in a large dataset.

We train this model by taking the CompCar [30] pretrained model as initialization, and perform training on our newly annotated AIC20 dataset. We obtain top-3 test accuracy of 73.90%, 6.05%, and 17.40%, for the vehicle type,

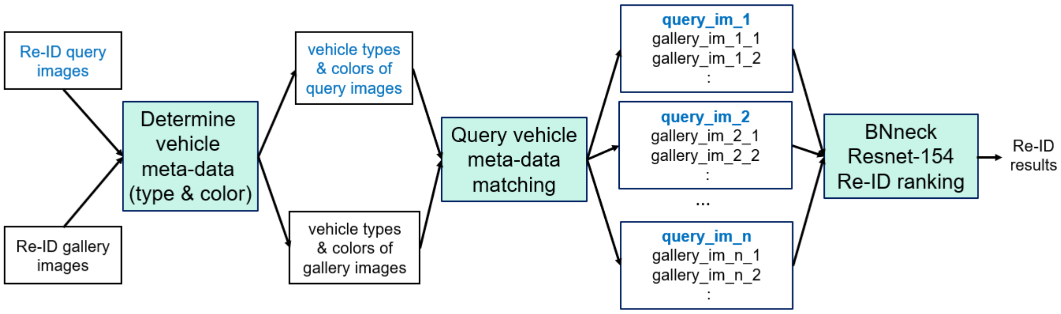
Re-ID query images
Re-ID gallery images
Determine vehicle meta-data (type & color)
vehicle types & colors of query images
vehicle types & colors of gallery images
Query vehicle meta-data matching
query_im_1
gallery_im_1_1
gallery_im_1_2
:
query_im_2
gallery_im_2_1
gallery_im_2_2
:
...
query_im_n
gallery_im_n_1
gallery_im_n_2
:
BNneck Resnet-154 Re-ID ranking
Re-ID results

Figure 3. **Track 2 vehicle re-identification** method overview.

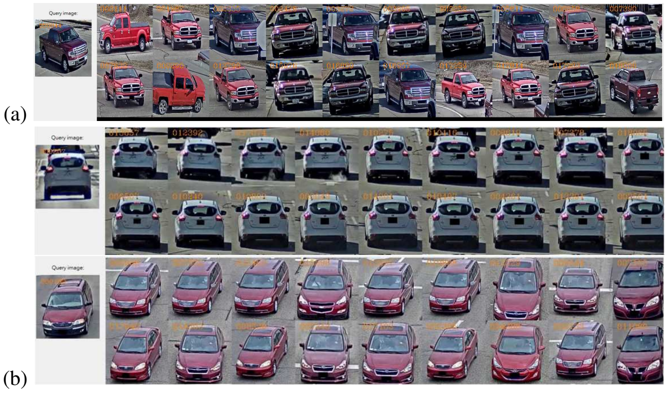

Figure 4. **Track 2 vehicle re-id** visual results. (a) Vehicle query results based on only using the vehicle type and color metadata. Observe that the accuracy of the car type is quite high, but in the color part, you can see that the light red, dark red, and dark red are all divided into the same among of the categories. (b) Final Re-Id query image (on the left) and the Re-Identified vehicles from the gallery images ranked ordered by similarities (row and column).

make, and color, respectively. This result shows that vehicle type and color are reliable features, however vehicle make is not reliable. This can be due to factors regarding data quality, as we observed that in the AIC20 dataset, many vehicle makes are hard to identify even from human eyes. Fig. 4(a) shows an example of vehicle type and color classification.

3.2.3 Pyramid Granularity Attentive Model

We use the Pyramid Granularity Attentive Model (PGAM) [2], a method we developed in AIC19 participation for vehicle re-id contest. The design of PGAM integrates the advantages of two recent deep re-id networks, namely the Multiple Granularity Model (MGN) [24] and Region-Aware deep Model (RAM) [12]. MGN is originally designed for person ReID, where the model contains a multi-branch deep network architecture. Specifically, MGN consists of a network branch for global feature representations, and two other branches for local feature representations. RAM operates similarly for re-id comparison, however several paths are used to deal with global and local features. PGAM is based on a ResNet-152 [7] backbone for better performance.

Specifically, to effectively focus the re-id learning on class centers in PGAM, we use the loss L_{cen} to tighten the clustering of learned class features. Losses L_{id} and L_{color} consider the summation of the three branches of losses regarding vehicle ID and color classification. Our vehicle PGAM re-id loss L_{reid} is defined as:

$$L_{reid} = \alpha_1 \ (L_{id} + L_{color}) + \alpha_2 \ L_{cen} + \alpha_3 \ L_{tri}, \qquad (3)$$

where L_{tri} denotes the standard triplet loss; weights α_1 , α_2, and α_3 control the importance among loss terms.

The training of PGAM takes the metadata-classified vehicle sets (*i.e.* vehicles of the same types and colors) as input. At run time, PGAM generates 1052 ranking result for each re-id query. Fig. 4(b) shows the example results. Observed that the ranking in Fig. 4(b) is more accurate and consistent compared to the raw metadata classification results in Fig. 4(a).

3.3. (T3) City-Scale Multi-Camera Vehicle Tracking

The AIC20 Track 3 challenge videos are captured by multiple concurrent cameras (as shown in Fig. 5). Difficulties arisen from the large variation of image qualities, viewing angles, occlusions, weather conditions, and the vast amount of potential vehicles to consider. How best to perform effective space-time tracklet fusion across cameras is the key problem to consider here. Our approach consists of three components:

(1) *multi-target single-camera tracking* (§ 3.3.1),
(2) *across-camera tracklet association filtering* (§ 3.3.2),
(3) *multi-camera tracking and re-id linking* (§ 3.3.3).

In step (1), the single-camera tracking considers both vehicle appearance and space-time features on the global (longitude, latitude) GPS coordinates. Steps (2) and (3) can be treated as the vehicle re-identification problem addressed in the AIC20 Track 2 challenge. However here the consideration can possibly extend to re-id of the whole vehicle tracks. In addition, here the space-time fusion of vehicles along the traffic flow provides an important cue. Vehicles moving along a traffic cue in most cases simply follow the

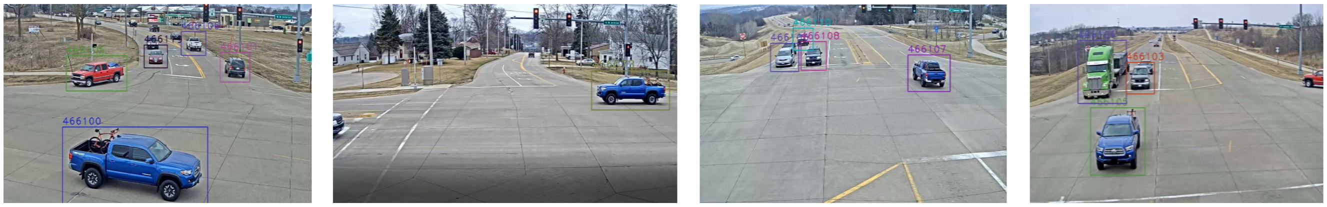

Figure 5. **Track 3 city-scale multi-camera vehicle tracking** examples.

traffic — lane switching or turning happen only occasionally, not all the time. So in step (2) we focus our approach on filtering out unlikely pair of vehicle trajectories before we feed them into the re-id network for considering trajectory linking. In step (3), we developed a new Multi-Camera Tracking Network (MCTN) that can effective link tracklets across camera views, by optimizing both physical and appearance cues.

3.3.1 Multi-target single-camera tracking

The AIC20 contest provides 3 vehicle detection results, namely Mask-RCNN [6], SSD512 [11], YOLOv3 [19]. We use Mask-RCNN detections as feed to our tracker. We adopt the leading method of [10] in AIC19 as our single-camera tracking method. Similar to the pipeline described in § 3.1 for vehicle tracking for counting, we perform Hungarian matching upon the combination of criteria (appearance re-id features and GPS spatial features), to find the matching of detection box pairs for vehicle tracklet formation. Vehicle GPS coordinates are obtained via projective transformation of camera views using the camera calibration matrices provided by the AIC20 contest. Vehicle appearance features are extracted using the re-id model based on the ResNet50 backbone. The re-id model is trained using the AIC20 Track 2 and Track 3 datasets.

3.3.2 Across-camera tracklet association filtering

To maintain a manageable multi-camera vehicle tracking (*i.e.* tracklet linking) across a large, cit-scale camera network, the aim here is to reduce the necessary amount of across-camera tracklet comparisons. We use two metrics to filter out vehicle tracklet pairs (T_p, T_q) that can be safely omitted.

The first cue we use is the physical (space-time) constrain between any pair of vehicle tracklets. Since all AIC20 Track 3 challenge videos are given with synchronized timestamps, from tracking we know the precise space-time localization of each tracklet, we can calculate the vehicle speed in MPH for each tracklet. This speed estimation together with the known distance between (T_p, T_q), we can estimate arrival time. In other words, suppose the tracklet pair (T_p, T_q) belongs to the same vehicle, we can estimate the arrival time based on the putative pair of vehicle tracklets using extrapolation. We denote t_a the *estimated arrival time* between (T_p, T_q), and t_b the the *time difference* between the first frames of (T_p, T_q). The difference $\delta_t = |t_a - t_b|$ can be used to filter out unlikely vehicle pairs across cameras:

$$\text{if } \delta_t > \tau_t, \text{ remove tracklet pair } (T_p, T_q), \tag{4}$$

where τ_t is set to 90 (frames).

We further filter out vehicle tracklet pairs (T_p, T_q) that travel in opposite directions, which is less likely to occur in a traffic flow. Specifically, let $vector_p$ and $vector_q$ denote the first and last sample points of two tracklets T_p and T_q, respectively. We consider the dot product (angle) between the vectors (v_p, v_q):

$$\text{if } v_p \cdot v_q < 0, \text{ remove tracklet pair } (T_p, T_q). \tag{5}$$

3.3.3 Multi-camera tracking and re-id linking

We next describe our Multi-Camera Tracking Network (MCTN) model developed for the AIC19 Track 3 challenge. We adopt the leading method of [10] from AIC19 to perform multi-camera tracking and re-id linking here. We found that the original method in [10] relies on many handcrafted threshold parameters for the cosine distance feature comparison for tracklet linking. To make the approach less problem-dependent, we use a data-driven approach to train a two-branch network in Fig. 6, with an aim to process the appearance features and physical features separately. The advantage of our design is that the 5-dim physical weights in the second branch can be learned independently and thus not affected by the 4096-dim appearance feature channel.

The 4096-dim **appearance feature branch** of MCTN (Fig. 6) consists of five fully-connected (FC) layers. Given a pair of tracklets (T_p, T_q), this network branch takes the pair of tracklet-average appearance features as input. Batch normalization is used in every hidden FC layers.

The 5-dim **physical feature branch** of MCTN (Fig. 6) consists of three fully connected layers. We use five physical features, namely three *vehicle GPS distances* d_1^{gps}, d_2^{gps}, d_3^{gps}, and two *vehicle timestamp distances* d_1^{ts}, d_2^{ts} as the branch input. Specifically, d_{gps1} denotes the Euclidean distance between the first sample points of (T_p, T_q) in GPS coordinates. Similarly, d_2^{gps} denotes the same quantity between (T_p, T_q), d_3^{gps} denotes the same between the last sample points of (T_p, T_q). The time difference terms d_1^{ts} and d_2^{ts}

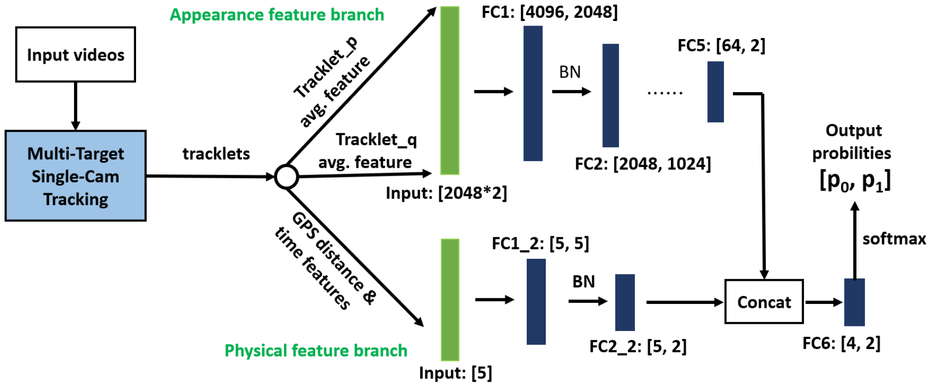

Figure 6. **Track 3 city-scale multi-camera vehicle tracking** using the proposed MTCN.

Table 1. **Track 3:** MTCN evaluation and ablation study results.

Model experiments	Accuracy
Only appearance branch	65%
Appearance + physical in a single branch	68%
MTCN appearance + physical branches	71%

denote the time difference between the first frames and last frames of (T_p, T_q), respectively.

The output of the above two branches are concatenated and fed to a FC layer. The output of MCTN is a 2D softmaxed vector indicating the (similarity, dis-similarity) probabilities between (T_p, T_q), which is denoted as (p_0, p_1), respectively. We use focal loss with an improved performance over cross entropy loss.

For the training of MTCN, we gather positive samples from vehicle tracklets of the same groundtruth IDs. We gather negative samples from different ID pairs. We keep the sizes of the both sets equal.

Table 1 shows the performance evaluation and ablation study of the proposed MTCN. We obtained 65% accuracy when using only the appearance features. After adding the physical features in a single branch, we obtained 68% accuracy. With the proposed two-branch design, we obtain 71% accuracy.

3.4. Traffic Anomaly Detection

Our traffic anomaly approach is based on a simple assumption that, in most cases after an anomaly condition occurs, the vehicles will not move and stay stationary on the road. Thus, the background extraction and smoothed averaging method are effective methods that can analyze the traffic flow to obtain road segmentation mask as well as the continuous stationary region (*i.e.* the vehicle of anomaly). We use a segmentation mask to filter out the "non-road" regions, such that we can perform anomaly detection without interference. In order to obtain the more continuous stationary region and background region, the MOG2 [31] and the smoothed averaging method [1] are used and combined their results. We use the Hybrid Task Cascade network [3] with ResNeXt backbone [28] as our vehicle detector, and fine-tune it on the AIC20 T4 vehicle detection training sets. Fig. 7 overviews our traffic anomaly detection pipeline.

We observed that in the traffic scene of the AIC20 Traffic Anomaly Detection Dataset (which consists of 100 training videos and 100 test videos), there exists many small vehicles in the distant which cannot be detected. This leads to miss-detected anomaly events. To address this problem, we use the perspective method in [1] to scale the small targets in the distant. This way, small vehicles will be maintained in a consistent and manageable size for applying our vehicle detector network.

After the position and the time information of the continuous stationary vehicles are obtained, we found that not all detected stationary vehicles are abnormal. For example, there are several vehicles waiting in the red light, which belongs to false-positive detections. Furthermore, the requirements in the T4 Challenge in accurately determining the anomaly starting time for the detected anomaly event is difficult in general. We apply the spatial-temporal matrix discrimination from [1] to determine the start time of the anomaly event.

4. Challenge Evaluation and Results

4.1. (T1) vehicle counts by class at multiple intersections

The AIC20 Track 1 challenge (vehicle counts by class at multiple intersections) provides 9 hours videos captured from 20 different vantage points. Video views include intersections of single ways, full intersections, highway segments, and city streets, covering various lighting and weather conditions (including dawn, rain, and snow). Videos are split into two data sets A and B. Data set A (5 hours in total) along with all the corresponding instruction documents and a small subset of ground truth labels (for demonstration purpose) are made available to all participant teams. Data set B will be reserved for later testing use.

The ranking of AIC20 Track 1 challenge is justified us-

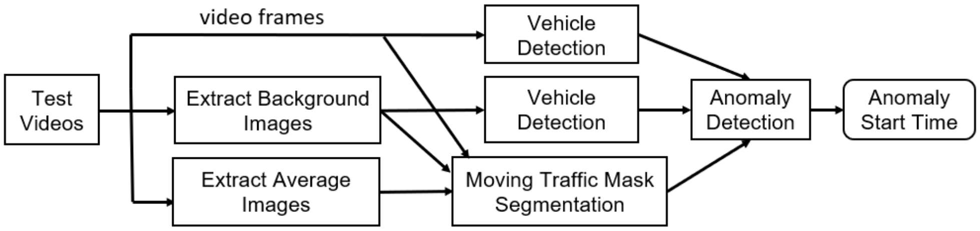

Figure 7. **Track 4 traffic anomaly detection** method overview.

ing both *vehicle counting accuracy* and running efficiency in terms of *execution time*. The vehicle counting accuracy calculates the counts of passing vehicles travelling along a set of pre-defined Movement of Interest (MOT) trajectories in the videos. The execution time measures how fast the test system process all tasks from the beginning to the end, using a given running efficiency python program (efficiency.py). Each participant team must execute this program to calculate an efficiency factor as part of submission results. The final evaluation score $S1$ is a weighted of effectiveness $S1_{effectiveness}$ and efficiency $S1_{efficiency}$ scores:

$$\alpha\ S1_{efficiency} + \beta\ S1_{effectiveness} \tag{6}$$

We obtained $S1$ score of 0.3241 which ranks **18-th** in the AIC20 general leaderboard. For the AIC20 public leaderboard, we obtained $S1$ score of 0.3116 with ranks **13-th**.

4.2. (T2) multi-camera vehicle re-identification

The AIC Track 2 contest (city-scale multi-camera vehicle re-identification) provides $56,277$ vehicle images, of which $36,935$ come from 333 vehicle identities form the training set and 18,290 from the other 333 identities in the test set. An additional $1,052$ vehicle images are used as queries in the evaluation. The vehicle re-id performance is evaluated by the standard mean Average Precision (mAP) of the top-K matches calculated from vehicle images in the query set, $K = 100$. The mAP is essentially the area under the Precision-Recall curve (PR-AUC) over all the queries. The AIC20 evaluation system calculates the mAP of the top-100 matches to rank the performance of each team.

We obtained mAP of 0.0368, which ranks **57-th** in the AIC20 general leaderboard (**41-th** in the AIC20 public leaderboard) out of all participant teams.

4.3. (T3) city-scale multi-camera vehicle tracking

The AIC Track 3 contest (city-scale multi-camera vehicle tracking) provides 215.03 minutes of videos collected from 46 cameras spanning 16 intersections in a mid-sized U.S. city. The multi-cam tracking performance of each participant team is evaluated using the F1 score of vehicle identity ($IDF1$), which measures the ratio of correctly identified detections over the average number of ground-truth and computed detections:

$$IDF1 = \frac{2TP_{id}}{2TP_{id} + FP_{id} + FN_{id}}, \tag{7}$$

where TP_{id} denotes THE identity true-positive, FP_{id} denotes the identity false-positive, FN_{id} denotes the identity false-negative.

We obtained $IDF1$ score of 0.0620, which ranks **6-th** out of the 9 world-wide participant teams in the AIC20 general leaderboards. We rank also **6-th** in the AIC20 public leaderboard out of 8 total teams.

4.4. (T4) traffic anomaly detection

The AIC20 Track 4 contest (traffic anomaly detection) data contains 100 training videos and 100 test videos in 800×410, each about 15 min long in 30 fps. These videos represent real-world traffic data covering large variety of traffic conditions, weather conditions (day, nights, snow, rainy, sunny), and traffic anomaly events (emergency stops, crashes).

Traffic anomaly detection is evaluated using the $F1$ score multiplied by the event detection time error in $RMSE$ (unit in seconds) as the final S_4 score:

$$S_4 = F1 \times (1 - NRMSE), \tag{8}$$

where the $NRMSE$ is the $RMSE$ normalized with minimum 0 and maximum 300.

We obtained F1-score of 0.9706, RMSE 6.6058, S4 94.92%, which ranks **4-th** out of 13 world-wide participant teams in the AIC20 general leaderboard.

5. Conclusion

We presented methods and results for our participation to all four contest tracks of the AI City Challenge 2020: on (T1) multi-class multi-movement vehicle counting, (T2) city-scale multi-camera vehicle re-identification, (T3) city-scale multi-camera vehicle tracking, and (T4) traffic anomaly detection, respectively. For T2 challenge, our newly annotated vehicle type and color classification dataset will share to the community that should enrich the available dataset. We achieved top-6 and top-4 ranking on the AIC20 general leaderboard for the T3 and T4 contests, respectively. **Future work** includes the continue improvement of the proposed method, as well as improving the execution speed running on edge devices.

References

[1] Shuai Bai, Zhiqun He, Yu Lei, Wei Wu, Chengkai Zhu, and Ming Sun. Traffic anomaly detection via perspective map based on spatial-temporal information matrix. In *CVPR Workshop on AI City 2019 Challenge*, 2019. 2, 3, 7

[2] Ming-Ching Chang, Jiayi Wei, Zheng-An Zhu, Yan-Ming Chen, Chan-Shuo Hu, Ming-Xiu Jiang, and Chen-Kuo Chiang. AI City Challenge 2019 – City-scale video analytics for smart transportation. In *CVPR Workshop on AI City 2019 Challenge*, 2019. 2, 4, 5

[3] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In *CVPR*, 2019. 7

[4] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ECO: Efficient convolution operators for tracking. In *CVPR*, pages 6638–6646, 2017. 2

[5] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, pages 4310–8, 2015. 2

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, pages 2961–2969, 2017. 2, 3, 6

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[8] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE PAMI*, 37(3):583–596, 2015. 2

[9] Tsung-Wei Huang, Jiarui Cai, Hao Yang, Hung-Min Hsu, and Jenq-Neng Hwang. Multi-view vehicle re-identification using temporal attention model and metadata re-ranking. In *CVPR Workshop on AI City 2019 Challenge*, 2019. 3, 4

[10] Peilun Li, Guozhen Li, Zhangxi Yan, Youzeng Li, Meiqi Lu, Pengfei Xu, Yang Gu, and Bing Bai. Spatio-temporal consistency and hierarchical matching for multi-target multi-camera vehicle tracking. In *CVPR Workshop on AI City 2019 Challenge*, 2019. 2, 3, 6

[11] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *ECCV*, pages 21–37, 2016. 6

[12] Xiaobin Liu, Shiliang Zhang, Qingming Huang, and Wen Gao. Ram: A region-aware deep model for vehicle re-identification. In *ICME*, pages 1–6, 2018. 5

[13] Kai Lv, Heming Du, Yunzhong Hou, Weijian Deng, Hao Sheng, Jianbin Jiao, and Liang Zheng. Mulvehicle re-identification with location and time stamps. In *CVPR Workshop on AI City 2019 Challenge*, 2019. 3

[14] Hyeonseob Nam, Mooyeol Baek, and Bohyung Han. Modeling and propagating CNNs in a tree structure for visual tracking. *arXiv:1608.07242*, 2016. 2

[15] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, pages 4293–4302, 2016. 2

[16] Milind Naphade, David C Anastasiu, Anuj Sharma, Vamsi Jagrlamudi, Hyeran Jeon, Kaikai Liu, Ming-Ching Chang, Siwei Lyu, and Zeyu Gao. The nVidia AI city challenge. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 1–6. IEEE, 2017. 1

[17] Milind Naphade, Ming-Ching Chang, Anuj Sharma, David C. Anastasiu, Vamsi Jagarlamudi, Pranamesh Chakraborty, Tingting Huang, Shuo Wang, Ming-Yu Liu, Rama Chellappa, Jenq-Neng Hwang, and Siwei Lyu. The 2018 NVIDIA AI City Challenge. In *CVPR Workshop on AI City 2018 Challenge*, pages 53—60, 2018. 1

[18] Milind Naphade, Zheng Tang, Ming-Ching Chang, David C. Anastasiu, Anuj Sharma, Rama Chellappa, Shuo Wang, Pranamesh Chakraborty, Tingting Huang, Jenq-Neng Hwang, and Siwei Lyu. The 2019 ai city challenge. In *CVPR Workshop on AI City 2019 Challenge*, 2019. 1, 2, 3

[19] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. 2018. 2, 3, 6

[20] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. *CVPR*, 2019. 1, 4

[21] Zheng Tang, Gaoang Wang, Hao Xiao, Aotian Zheng, and Jenq-Neng Hwang. Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features. In *CVPR Workshop on AI City 2018 Challenge*, pages 108–115, 2018. 2

[22] Liang Tong and Zhufang Li. Study on the road traffic survey system based on micro-ferromagnetic induction coil sensor. *Sensors & Transducers*, 170:73–79, 2014. 3

[23] Gaoang Wang, Xinyu Yuan, Aotian Zhang, Hung-Min Hsu, and Jenq-Neng Hwang. Anomaly candidate identification and starting time estimation of vehicles from traffic videos. In *CVPR Workshop on AI City 2019 Challenge*, 2019. 3

[24] Guanshuo Wang, Yufeng Yuan, Xiong Chen, Jiwei Li, and Xi Zhou. Learning discriminative features with multiple granularities for person re-identification. In *ACM Multimedia*, pages 274–282, 2018. 5

[25] Longyin Wen, Zhen Lei, Ming-Ching Chang, Honggang Qi, and Siwei Lyu. Multi-camera multi-target tracking with space-time-view hyper-graph. *IJCV*, 122(2):313–333, 2017. 2

[26] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, pages 3645–3649, 2017. 2

[27] Xuzhi Xiang, Mingliang Zhai, Ning Lv, and Abdulmotaleb El Saddik. Vehicle counting based on vehicle detection and tracking from aerial videos. *Sensors*, 18(2560), 2018. 3

[28] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural network. In *CVPR*, 2017. 7

[29] Fu Xiong, Yang Xiao, Zhiguo Cao, Kaicheng Gong, Zhiwen Fang, and Joey Tianyi Zhou. Towards good practices on building effective cnn baseline model for person re-identification. *arXiv:1807.11042*, 2018. 3

[30] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*, 2015. 4

[31] Zoran Zivkovica and Ferdinandvan der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780, 2006. 7