Chapter

Vehicle Tracking Using Video Surveillance

Sandesh Shrestha

Abstract

In numerous applications including the security of individual vehicles as well as public transportation frameworks, the ability to follow or track vehicles is very helpful. Using computer vision and deep learning algorithms, the project deals with the concept of vehicle tracking in real-time based on continuous video stream from a CCTV camera to track the vehicles. The tracking system is tracking by detection paradigm. YOLOv3 object detection is applied to achieve faster object detection for real-time tracking. By implementing and improving the ideas of Deep SORT tracking for better occlusion handling, a better tracking system suitable for real-time vehicle tracking is presented. So as to demonstrate the achievability and adequacy of the framework, this chapter presents exploratory consequences of the vehicle following framework and a few encounters on handy executions.

Keywords: object detection, YOLOv3, multiple object tracking, Deep SORT, vehicle re-identification

1. Introduction

Innovation has changed the manner in which individuals convey and work with one another. Vehicle tracking frameworks, as the name recommend, permit the following of vehicles in its most fundamental capacity. They utilize a mix of innovations to keep ongoing tabs on the situation of a vehicle or to develop a background marked by where a vehicle has been. These frameworks are utilized in an assortment of enterprises, and they likewise a key piece of most stolen vehicle recuperation methodologies. Vehicle tracking is a significant innovation for the safety of the fleet of vehicles as well as for the conventional driver. This is to turn out to be much progressively significant for road safety as the innovation turns out to be progressively available and cheap. There are two kinds of vehicle tracking, every one of which is valuable in explicit circumstances.

Passive vehicle tracking: these trackers normally utilize a GPS gadget to record the situation of a vehicle after some time. At the point when the tracker is expelled, the information can be moved to a PC and broke down. These following frameworks are valuable for fleet management, yet they likewise have different applications.

Active vehicle tracking: increasingly intricate tracking frameworks transmit the area of a vehicle progressively. For fleet management and dispatch purposes, this information is normally observed from a focal area. This kind of framework can likewise be utilized for stolen vehicle recuperation.

The proposed system tracks vehicles esp. cars on the road or highway by constant surveillance by the CCTV cameras installed at a certain elevation (on the bridge). The object detection algorithm is run on each frame of the video stream obtained in real-time from the CCTV camera. If the object detection system detects a vehicle (car), the system starts tracking each detected vehicle on each frame. The proposed system can also be utilized in the autonomous vehicle industry to escort so particular vehicle. In light of the tracked vehicle, autonomous vehicles can make decisions.

In Computer Vision, tracking is one of the most significant fields. Tracking is the issue of estimating the position of an object over continuous image sequences. This is likewise additionally isolated into two subcategories-single object tracking and multiple object tracking. Both of them two require marginally various methodologies.

1.1 Challenges in tracking systems

It is very crucial to know the challenges we need to take care of during tracking. Some of the common and major challenges are as follows.

- *Object occlusion*: If the target object is occluded or blocked by other objects in a sequence of images, then it not only becomes difficult to detect the object but to update future images too if it becomes visible again.

- *Fast movement*: Cameras, such as on smart-phones, often suffer from jittery movement. This produces a blurred effect and, sometimes, the object becomes completely absent from the frame. Therefore, sudden changes in the motion of cameras also lead to problems in tracking applications.

- *Change of shape*: If we are targeting non-rigid objects, changes in the shape or the complete deformation of an object will often lead to being unable to detect the object and tracking failure.

- *False positives*: In a scene with multiple similar objects, it is hard to match which object is targeted in subsequent images. The tracker may lose the current object in terms of detection and start tracking a similar object.

These challenges can make our tracking application fail and give the wrong estimate of an object's location.

The bigger challenge is to deal with these problems in a real-time scenario. One of the ongoing accessible tracking frameworks is Simple Online and Real-time Tracking (SORT) [1] that attempted to beat these difficulties. It is a straightforward system to track people progressively. SORT uses the Kalman filter on each frame of the video. To solve the association problem in visual tracks, Hungarian calculation is utilized. But their proposed framework is appropriate only for tracking humans in various appearance scenes. This system still involves identity switch problem and does not handle occlusion.

Simple Online and Real-time Tracking with Deep Association Metric (Deep SORT) [2] is an improvement over SORT. It used appearance features from deep convolutional neural network (CNN) for handling occlusion during tracking the people. Deep SORT uses a cost matrix dependent on both motion information and appearance features from the CNN model to abstain from missing tracking as a result of tracking or missed detection of people. Their framework incorporates a convolutional neural system for an individual's evident highlights trained on

a person re-identification dataset. This framework also is applicable only for tracking humans.

This paper presents a unique technique for detecting vehicles and tracking them. Using motion features and appearance features, a distinctive approach is provided to track the vehicles in real-time. This framework utilizes a convolutional neural system to obtain the appearance features to track the vehicle (car). A separate vehicle re-identification database has been created to train with an improved CNN model.

1.2 Major contribution

The proposed system ameliorated the tracking performance in the MOT problem. It presents the visual tracking system for vehicles on the road using the similar principles used for human tracking. Given better equipment assets, the proposed framework beats the current best in class frameworks as far as accuracy and performance. The major contributions are as follows.

1. Improved real-time tracking performance by replacing Faster R-CNN by YOLOv3 which is a faster object detector with similar accuracy.

2. Created a vehicle re-identification dataset from scratch necessary to train the CNN model for obtaining appearance features.

3. Modified the CNN model of Deep SORT for better tracking performance.

The remainder of the paper is classified into various areas. Section 2 gives the detailed foundation of past methodologies for MOT frameworks. Section 3 describes the methodology alongside complete design. Segment 4 illuminates experimentation and evaluation. The last segment concludes the proposed framework for vehicle tracking.

2. Background

The research community has begun to focus on tracking every single object in distinct settings with the enhancement in multi-object detection. The entire MOT problem can be considered as an association problem in which the basic goal is to associate the objects detected. Following object detection using some object detection algorithm, tracking is implemented. In this segment, the following systems will be reviewed.

- Object detection algorithms since the first step is to detect the vehicles before tracking.

- Already available tracking systems.

2.1 Review of object detection systems

At the beginning of the 1990s, object detection was completed utilizing template matching algorithms [3], where a format or template of the particular item is slid over the information picture to locate the most ideal match in the info picture. In the late 1990s, geometric appearance-based object detection got the center of

attention [4, 5]. The essential spotlight was on the tallness, width, angles, and other geometric properties in these techniques.

During the 2000s, the object identification model was moved to low-level features dependent on some statistical classifiers, for example, local binary design (LBP) [6], oriented gradient histogram [7], scale-invariant feature transform [8], and covariance [9]. Detection and classification of objects based on extraction characteristics engaged machine learning based on extracted features.

Handcrafted traditional characteristics have been used for object detection for many years in the field of computer vision. But, with the advancement in deep learning following the notable results of the challenge of image classification in 2012 [10], CNNs are being used for this purpose. Researchers moved their attention to object detection and classification after the achievement of object classification in [10]. Deep convolutional neural networks operate extremely well in terms of edges, texture, and appearance to extract local and global features.

The research community has shifted in the latest years towards region-based object detection networks. In various apps such as video description [11], this sort of object detection is used. Convolutional characteristics are obtained over suggested areas in region-based algorithms for object detection, followed by region categorization into a particular class.

With the appealing results of AlexNet [10], Girshick et al. [12] suggested the concept of object detection using a convolutional neural network. They used a selective search to propose regions where you can find prospective objects [13]. They called their network of object detection as a neural network (R-CNN) area convolution. The fundamental flow of the neural network area convolution (R-CNN) can be defined as follows.

- Regions are suggested using the selective search [13] for each item in the input image.
- Proposed areas are resized to the same coherent size to classify the proposition into predefined classes based on regional CNN features obtained.
- The softmax layer was substituted by the linear SVM classifier to train the system on fixed-length CNN features.
- Ultimately, a bounding box regressor is used to locate the object perfectly.

Despite being a major breakthrough in the field of object detection, the suggested R-CNN has some important shortcomings.

- Training procedures are quite slow because R-CNN has to train separately in different phases.
- Selective searching, which is itself a slow method, proposes regions.
- It is expensive to train the separate SVM classifier as CNN features are extracted for each region, making SVM training even more challenging.
- Detection of objects is slow because CNN characteristics are extracted for each test image for individual suggestions.

Kaiming He et al. [14] suggested spatial pyramid pooling (SPP) to solve the problem of feature extraction for each proposition. The fundamental concept was to

recognize the input of any size by the convolutional layers; fully connected layers force input to be set size to make the multiplication of matrix feasible. Following the last convolutional layer, they used SPP layer to obtain the fixed size characteristics to feed into a fully connected layer. Comprehensively enhanced performance was obtained with SPPNet R-CNN. For suggestions of distinct dimensions, SPPNet extracts convolutional functions on the input picture only once. This network enhances testing efficiency, but it does not enhance R-CNN training performance. In addition, the weights of convolutional layers in front of the SPP layer cannot be altered which limits the method of fine-tuning.

R-CNN's primary contributor, Girshik [15], suggested that Fast-ECNN address some R-CNN and SPPNet issues. Fast R-CNN uses the concept of sharing convolutional computation for various suggested areas. It adds a region of interest (ROI)-pooling layer to create fix-sized features of individual proposals after the last convolutional layer. ROI-pooling layers' fix-size features are supplied to the stack of fully connected layers that further divided into two branch networks: one serves as the network for object classification and the other for bounding box regression. They asserted that R-CNN's general training step efficiency was improved by three times and that for testing was improved by ten times.

While Fast R-CNN has significantly enhanced R-CNN's efficiency, it still utilizes selective search as a regional proposal network (RPN). The Regional Proposal phase consumes the time that operates as the Fast R-CNN bottleneck. Modern advances in object location using deep neural networks [16] have driven Ren et al. [17] to use CNN to replace slow regional proposal processes using selective search. They suggested effective RPN proposals for objects. RPN and Fast R-CNN share, respectively, the convolutional layers for region proposal and region classification in Faster R-CNN. Faster R-CNN is a purely convolutional neural network with no handcrafted features that use a fully convolutional neural network (FCN) for region proposal. They asserted that for the test stage, Faster R-CNN could function at 5 fps.

Redmon et al. [18] suggested a new method for detecting objects called You Only Look Once (YOLO). The region proposal phase was totally dropped; YOLO divides the entire image into grids and predicts detection on candidate regions bases. YOLO splits the entire picture into the grids of S × S. Each grid has a probability of class C, B as the locations of the bounding box, and a likelihood for each box. Removing the RPN step enhances detection efficiency; YOLO can detect the objects while operating around 45 fps in real-time. YOLOv2 [19] and YOLOv3 [20] are the later improvements on YOLO. YOLOv3 processes images at 30 fps; as accurate as other state-of-art object detectors but 3× faster.

2.2 Review of multiple object tracking

Multiple scientists concentrated on motion and spatial characteristics to track multiple objects [21, 22]. To capture the associations between various detections [23, 24], some of the scientists concentrated on appearance features.

There are some traditional techniques that create a frame-by-frame prediction. These traditional methods require multiple hypothesis tracking (MHT) [25] and the JPDAF filter [26]. To track the identified items, both of these ancient methodologies involve a lot of computation. The complexity of these methodologies improves exponentially as the number of trackable objects rises, which makes them very slow to be used in complicated environments for online applications. A single state hypothesis is produced in JPDAF based on the relationship between individual measurement and probability of association. In MHT, consideration is given to a full set of hypotheses for monitoring followed by post pruning for tractability.

By offering JPDA approximation, Rezatofighi et al. [27] made an attempt to enhance the efficiency of JPDAF. They have utilized the latest advances in solving an integer program's m-best solution. This system's primary benefit is to make JPDA less complicated and more tractable. They redefined the technique for calculating an individual JPDAF assignment as a solution to a linear program. Another team of researchers, Kim et al. [23], used features based on appearance to track the target. By pruning MHT's graph to achieve state-of-the-art performance, they improved the MHT. They used the least regularized squares to increase the MHT methodology's effectiveness.

Compared to the legacy implementations, these two improvements perform quite well, but these two techniques still have a great deal of delay in the decision-making phase making these techniques unsuitable for real-time applications. By raising the individual density, these techniques involve big computational resources.

Some scientists have been working on the theory of graphs to track people. Kayumbi et al. [28] suggested a multi-camera viewing algorithm to discover the trajectories of football players based on distributed sensing algorithms. Their algorithm begins with mapping the camera view plane to the floor plane's virtual top perspective. Finally, to track each person on the ground plane, they utilized graph theory.

Some online tracking techniques use individual appearance features to track [29, 30]. These models extract apparent individual look features. Both schemes provide precise descriptors of the appearance to guide the association of data. The first system incorporates individuals' temporal appearance along with the features of the spatial appearance. After tuning the parameters in each iteration, their appearance model is learned by implementing the incremental assessment. Markov decision process (MDP) is used in the second system to map the detected object's age in Markov chain terms. MDP determines the tracks based on the target's present status and history.

Some of the scientists have recently been working on simple online tracking and trying to create live stream tracking in real-time [1, 2]. These schemes are referred to as simple online and real-time tracking, and simple online and real-time tracking with a deep metric association. Two consecutive variants of the same methodology are both of these schemes. Kalman filter is used in both schemes to discover the target's motion features. These systems used as the key tracking parameters like intersection over union, central position, height, width, and velocity. Convolutional features for target appearance are also used in Deep SORT along with movement features to decrease missing tracks in various frames after occlusions and missed detections. This framework has been used only for human tracking.

The proposed approach improves the performance of tracking by redefining the CNN model to decrease the number of identity switches and applies it to track vehicles in real-time.

3. Methodology and framework

Deep SORT is tracking by detection paradigm. In tracking by detection, each frame uses an object detector to find possible instances of objects and then matches those detections with corresponding objects in the preceding frame. In the MOT problem setting, there are many objects to track in each frame. There are two steps to a generic technique of solving this-Detection and Association. In real-world applications, prior bounding box detections are necessary, so tracker could be combined with a detector. First, all the objects in the frame are detected. Single or multiple detections may be available. Once we have frame detections, comparable

detections are matched with the prior frame, to get the tracking for an object, the matched frames are carried through the sequence.

This generic method can be further divided into three steps.

1. A technique of identification of objects based on CNN is used to calculate detections. Faster-RCNN is used in the Deep SORT paper [2] to conduct the initial detection on every frame. This scheme uses YOLOv3.

2. An estimation model is an intermediate phase before the data association. This utilizes the status of each track as an eight-quantity vector, i.e., bounding box center (x, y), box height (h), box aspect ratio (γ), and its derivatives with time as velocities. The Kalman filter is used to model these states as a dynamic system. If for a limit of successive frames there is no detection of a tracking object, it is regarded to be out of frame or lost. The new track is initiated for a newly detected box. In addition, the bounding box descriptors are computed using a pre-trained CNN.

3. In the final step, given the predicted states from Kalman filtering using the previous information and the newly detected box in the current frame, an association is made for the new detection with old object tracks in the previous frame. This is computed using Mahalanobis distance between the detection and the position of the track predicted by the Kalman filter and Cosine distance obtained from the appearance feature set.

3.1 System flowchart

The tracking procedure consists of many complex concepts and calculations. All the processes involved are explained below in detail.

3.1.1 State estimation

The tracking scenario is defined on eight-dimensional state space (u, v, γ, h, $\dot{x}$, $\dot{y}$, $\dot{\gamma}$, $\dot{h}$) where (u, v) is the bounding box position, h is the height and γ is the aspect ratio and their respective velocities in image coordinates. A standard Kalman Filter is used with constant velocity motion and a linear observation model is used to solve the velocity components.

3.1.2 Trajectory processing

This mainly says when the trajectory terminates and when a new trajectory is generated. First, for each track, there is a threshold of 30 frames for recording the time from the last successful match to the current time. When the value is greater than the threshold set in advance, the track is deleted. A new trajectory may be generated for the detectors that have no matching success. However, since these detections may be some false alarms, the newly generated trajectory is marked with the state 'tentative'. Then it is observed whether the consecutive matching is successful in the next 3 consecutive frames. If so, it is considered as a new trajectory, marked as 'confirmed', otherwise, it is considered a false trajectory, and the status is marked as 'deleted'.

3.1.3 Motion matching

Matching naturally refers to the match between the currently active track and the current detection. The so-called effective trajectory refers to the trajectories that

are still alive, i.e., the trajectories whose states are tentative and confirmed. The degree of motion matching is plotted using the Mahalanobis distance between the detection and the position of the track predicted by the Kalman filter. The Mahalanobis distance takes into account state estimation uncertainty by assessing how many standard deviations the detection is away from the mean track location.

$$d^{(1)}(i,j) = \left(d_j - y_i\right)^T S_i^{-1}\left(d_j - y_i\right) \tag{1}$$

indicates the motion matching degree between the jth detection and the ith trajectory, where S_i is the covariance matrix of the observation space at the current time predicted by the Kalman filter. (y_i, S_i) is the projection of the ith track distribution into measurement space and d_j is the jth bounding box detection.

The 0.95 quantile of the inverse Chi-square distribution is used as the threshold for the Mahalanobis distance to exclude unlikely associations which is denoted by

$$b^{(1)}(i,j) = (1)\left[d^{(1)}(i,j) \le t^{(1)}\right] \tag{2}$$

that evaluates to 1 if the association between the ith track and jth detection is admissible. The corresponding Mahalanobis threshold for four-dimensional measurement space is $t(1) = 9.4877$.

3.1.4 Appearance matching

The use of the Mahalanobis distance matching metric alone can lead to serious conditions such as ID Switch, especially when the camera motion may cause the Mahalanobis distance metric to be invalid, so this time should be remedied by apparent matching. For each detection, including the detections in the track, the deep network is used to extract the feature vector. A gallery is built for each tracking target, storing the feature vectors of the last 100 frames that each tracking target successfully associates with. The second metric is calculated as the minimum cosine distance between the feature set of the last 100 successful associations of the ith track and the feature vector of the jth detection result of the current frame.

$$d^{(2)}(i,j) = min\left[1 - r^T r_k^{(i)} | r_k^{(i)} \varepsilon R_i\right] \tag{3}$$

where r is the appearance descriptor and R is the gallery.

If the above distance in Eq. (3) is less than the specified threshold, then the association is successful. The threshold is obtained from a separate training set.

$$b^{(2)}(i,j) = (1)\left[d^{(2)}(i,j) \le t^{(2)}\right] \tag{4}$$

To build the association problem, both metrics are combined using a weighted sum

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j) \tag{5}$$

where an association is admissible if it is within the gating region of both metrics

$$b_{i,j} = \prod_{m=1}^{2} b_{i,j}^m \tag{6}$$

The influence of each metric on the combined association cost can be controlled through the hyperparameter λ. For the situations where camera motion is present, set $\lambda = 0$.

3.1.5 Deep appearance descriptor

Person re-identification in video surveillance is a familiar issue in which a given query image is used to look at a huge gallery of images that have been gathered at distinct moments, lighting environments from different cameras that may contain the same individual. Direct classification is discouraged in this situation because the training set does not include people in the gallery gathered at the moment of the test. Thus, the re-identification model is used in which the goal is to learn a feature representation-from a separate training identities-suitable for performing nearest neighbor queries on images and identities provided at test time. The primary concept of a simple re-identification model is that the output of the CNN model without the last fully linked layer can give us the feature of a person image. Then we can calculate the cosine similarity of two features of the individual to get whether or not these two are the same individual.

The overview of the network architecture of the re-identification model used in Deep SORT framework is as follows (**Figure 1**).

The architecture of the network is shallow to promote quick training and to apply online tracking that involves quick computation of appearance features. Input images in RGB color space are rescaled to 128×64 and fed to the network. A series of convolutional layers then reduce the size of the feature map to 16×8 before fully connected layer Dense 10 extracts a global feature vector of 128. There are six residual blocks in the network. All the convolutions are 3×3 and the layer of the Max Pool is 2. Dropout and normalization used as regularization means. The Dense layer is introduced at a stage where the feature map still offers sufficient spatial resolution for re-identification. Before the application of the softmax classifier, *l2* normalization projects features on the unit hypersphere after the final Dense layer. Exponential Linear Units (ELUs) used as activation function and Cross-Entropy used as the loss function.

In this approach, the network architecture was modified to get more features and to increase the accuracy of the trained model which in turn would improve the tracking efficiency. The modified CNN architecture of the re-identification model is as follows (**Figure 2**).

Various architectures were tested, but this architecture resulted in maximum accuracy. In this model, one of the convolutional layers before Max Pooling is removed and eight residual blocks are used before Average Pooling to get 512 features. Then the Dense layer converts it into 256 feature vectors before applying *l2* normalization. The addition of many layers also degraded the performance because by going deep, the feature maps couldn't provide enough spatial resolution. Instead of ELU, the Rectified Linear Unit (ReLU) activation function was used in this model.

3.1.6 Cascade matching

When a target is occluded for a long time, the uncertainty of the Kalman filter prediction is greatly increased and the observability in the state space is greatly reduced. In a case where two tracks compete for the matching with the same detection, the track that is occluded for a longer period often has more uncertainty in tracking the predicted position because the position information is not updated for a long time, i.e., the covariance is larger. The Mahalanobis distance calculation

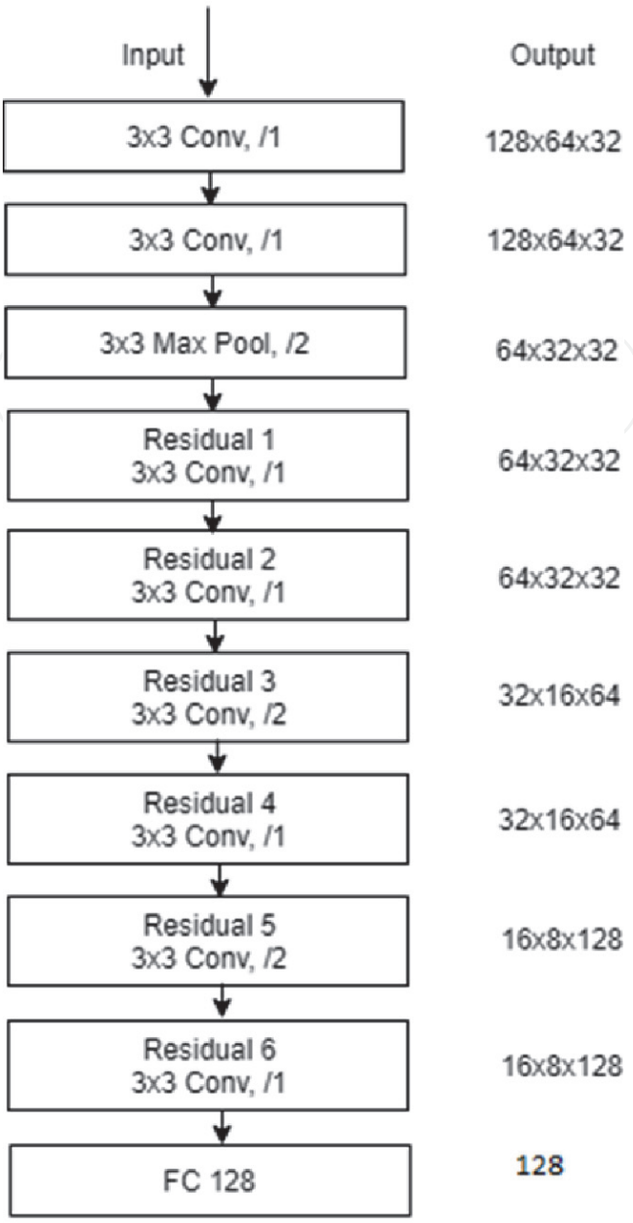

Figure 1.
The re-identification CNN architecture of Deep SORT framework.

uses the reciprocal of the covariance, so for large covariance, the Mahalanobis distance is smaller, so the detection result is more likely to be associated with the trajectory with a longer occlusion time. This undesired effect often destroys the continuation of the tracking. Thus, cascading matching is used to give priority to more frequently occurring targets.

In cascade matching, the set of indices of track T and detection D as well as the maximum age of A_{max} are provided as input. We calculate the price matrix of the association and the matrix of admissible associations are calculated according to Eqs. (5) and (6). Then iteration is done over track age n to solve a linear assignment problem for increasing age tracks. Next, a subset of T_n tracks not connected with the last n frames detection are picked. After that, the linear assignment is solved between T_n tracks and unmatched U detections. Finally, the set of matches and unmatched detections are updated. This matching cascade provides priority to lower age tracks, i.e., tracks that have recently been seen.

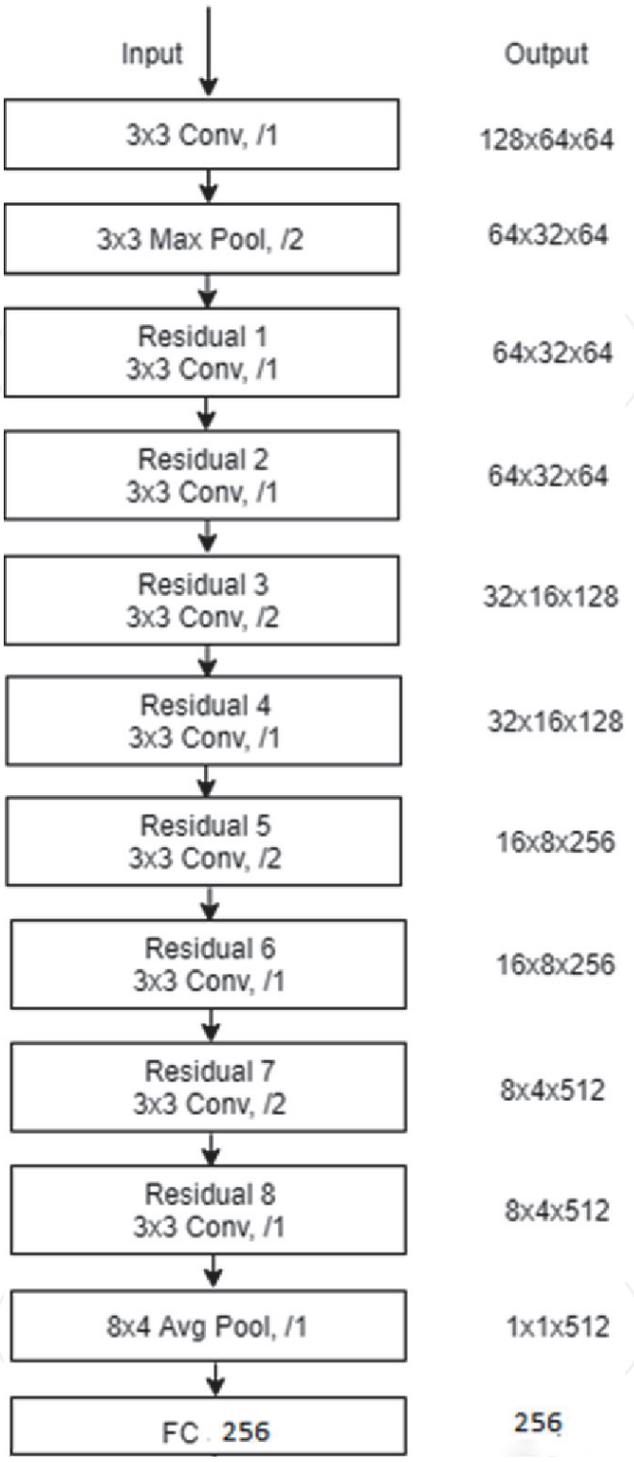

Figure 2.
The modified CNN architecture of re-identification model.

3.1.7 Intersection over union (IoU) matching

IoU-based matching of unconfirmed and age = 1 unmatched trajectories is also performed at the final stage of the match. This can alleviate large changes due to apparent mutations or partial occlusions. Of course, there are advantages and disadvantages. This may also cause some newly generated tracks to be connected to some old tracks. But this is less.

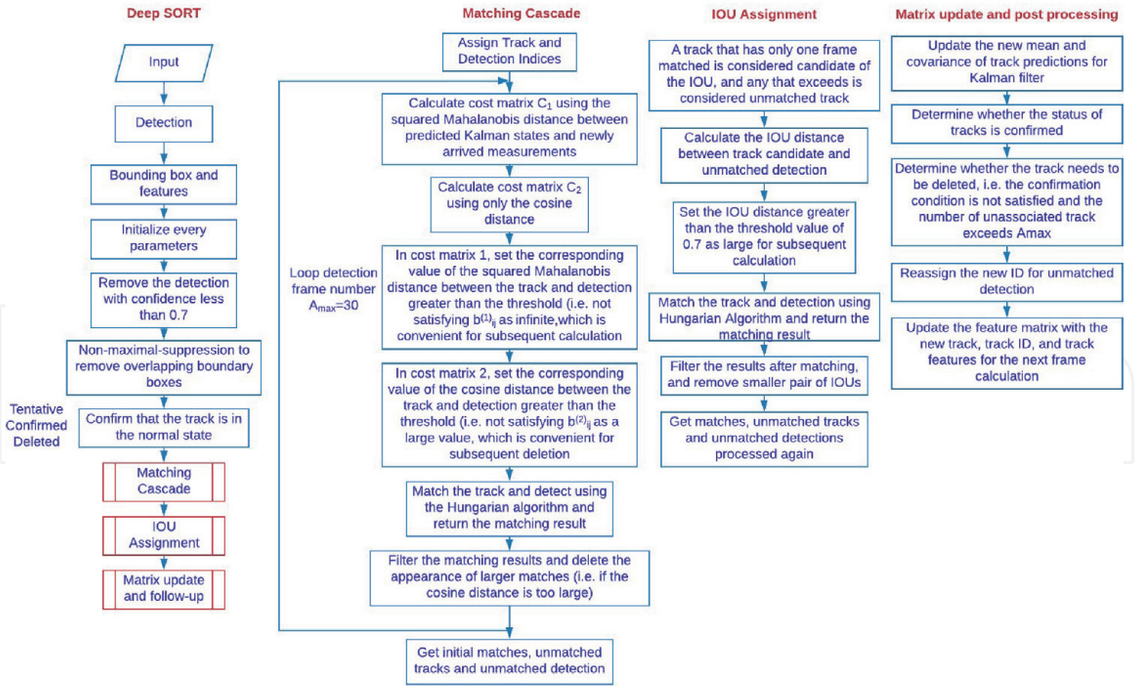

Figure 3.
Flowcharts involved in the tracking process.

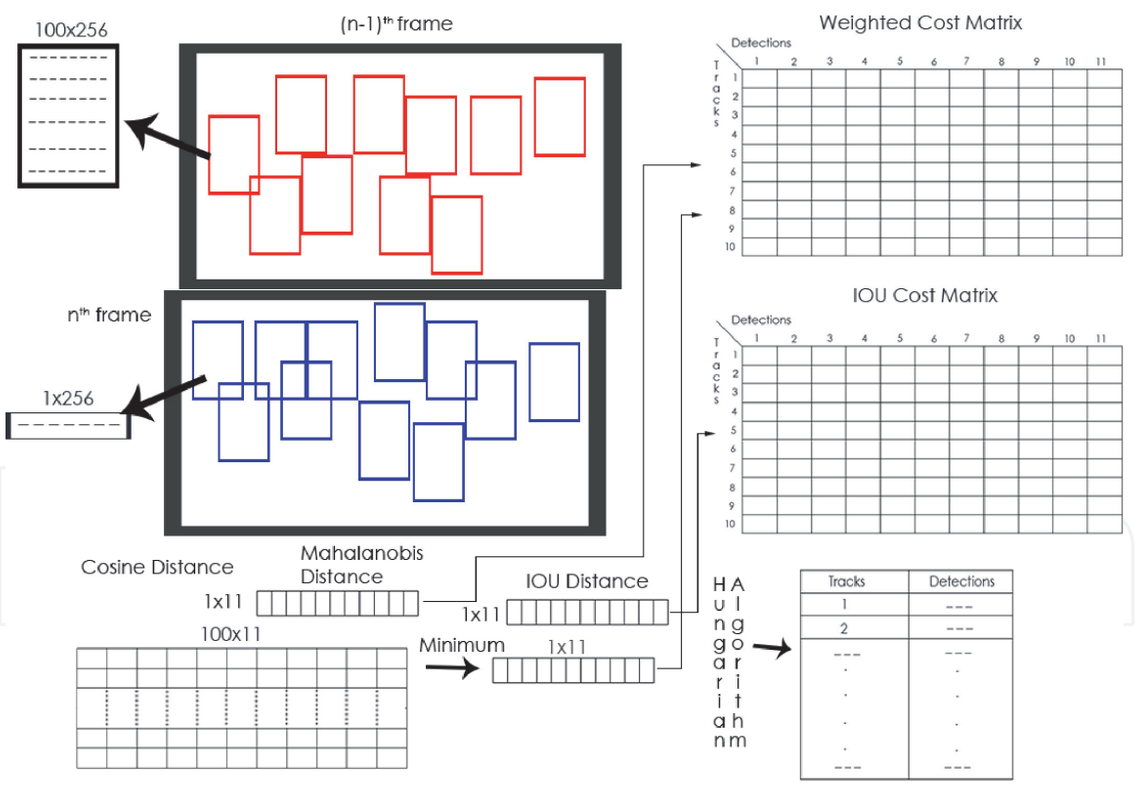

Figure 4.
Pictorial representation of matching during the tracking process.

The complete system flowchart is explained as follows (**Figures 3** and **4**).

1. Read the position of each detected object in each frame and the feature of each detection.

2. Filter the detection frame according to the confidence level, i.e., delete the detection frame and features with insufficient confidence.

3. Perform non-maximum suppression on the detection frame to eliminate multiple frames on one target.

4. *Prediction:* Use Kalman filtering to predict the position of the target in the current frame.

Perform Kalman filter equations:

$$x(k) = Ax(k-1) \tag{7}$$

and

$$p(k) = Ap(k-1)A^T + Q \tag{8}$$

where $x(k)$ is status information for the target (mean), $x(k\ 1)$ is the information of the target in the previous frame [center x, center y, aspect ratio, height, 0, 0, 0, 0]; $p(k\ 1)$ is the estimated error of the target (covariance); A is the state transition matrix; Q is the system error or noise matrix.

5. *Update:* Update the Kalman tracker parameters and feature set, and additionally assess the target disappearance and the new target appearance.

a. Match results with tracking prediction results.

- A tracker must distinguish between confirmed states and unconfirmed the status.
- Tracker with confirmed status is set for cascading matching.
- For a plurality of trackers of the same disappearance time, calculate a cosine distance matrix between the depth feature of each target newly detected by the current frame and the feature set saved by each tracker. If there are 11 detection targets in the current frame, there are 10 trackers, and each tracker has retained the depth features (256 features) of the previous 100 frames. The calculated cost matrix size is 10×11, and the calculation process is first for each tracker. For 100 features, calculate (1 − cosine distance) between the 11 new detection target features of the current frame, get 100×11 matrix, then find the minimum cosine distance for each detection block, get 1×11 matrix, and store it in cost matrix corresponding row, indicating the minimum cosine distance between the current tracker and the current detection block.
- Calculate the Mahalanobis distance between the predicted position of the Kalman filter and the detection frame. The specific process is to convert each detection frame from [x, y, w, h] to [center x, center y, aspect ratio, height]. For each tracker, i.e., each row in cost matrix, calculate the Mahalanobis distance between the prediction result and the detection result. Assuming the frame has 11 detection results, there is a matrix with the distance of 1×11, for the current row in the cost matrix. The position where the Mahalanobis distance is greater than the specified threshold is assigned as 1e+5.
- Set the element larger than the max distance in cost matrix to cost matrix > max distance.

- Assign the track to the correct detection by minimum cost matching of the cost matrix using the Hungarian algorithm.

- After the assignment is completed, the unassociated detections and tracks are regarded as unmatched detections and unmatched tracks. The matching whose cost matrix value is greater than max distance (threshold) is also sent to the unmatched detections and the unmatched tracks.

- Further matching of unconfirmed and unmatched tracks is based on the IoU. The specific implementation is to calculate the cost matrix. The elements larger than the max distance in cost matrix are set to the max distance, and then the Hungarian algorithm is used to assign the cost matrix as input. After the assignment is completed, the outputs are still matched, unmatched detections, unmatched tracks.

b. *To do parameter updates:*

- The process of parameter updating is to calculate the Kalman filter formulas

$$K(k) = p(k)H^T\left(Hp(k)H^T - R\right) - 1 \tag{9}$$

$$x(k) = x(k-1) + K(k)(z(k) - Hx(k-1)) \tag{10}$$

$$p(k) = (1 - K(k)H)p(k-1) \tag{11}$$

where $K(k)$ is the Kalman gain, z is the measurement vector, H is the measurement matrix and R is the measurement noise.

- After the parameter update is completed, the feature is inserted to the tracker feature set and corresponding parameters are re-initialized.

a. *Delete unmatched tracker:* The unmatched tracker indicates that although the new location is predicted, the detection box does not match.

b. *Initialize unmatched detection to a new tracker:* If there is no matched detection, indicating that a new target to be tracked appears, at this point, a new Kalman filter is initialized, and a new tracker is initialized.

c. Delete the trackers to be deleted.

d. Update the feature set of the remaining trackers.

3.2 Dataset preparation and training

To create the vehicle re-identification dataset, thousands of pictures of vehicles on the road were collected. A CCTV camera was placed at the bridge which captured the images of the vehicles on the road below the bridge. Over 33,000 images of 600 different vehicle models were collected. The collected vehicle images were cropped into 128 × 64 pixels. The training and test sets were split into 80 and 20% respectively and both were structured as follows by placing all the images of one vehicle into a specific numbered folder for both training and test sets. The sample of the images used in the dataset is shown below (**Figure 5**).

The system was run on the hardware with the following configurations.

CPU: Intel Core i7-7770k, 4.3 GHz
Motherboard: Asus Prime Z270-A and Asus ROG-SYRIX-GTX1080TI
RAM:16 GB

The experimentation steps were performed using Python programming language with PyTorch framework.

The following configurations were used for training both the original re-identification model and the modified re-identification model.

batch size = 64
no. of epochs = 40
learning rate = 0.1
momentum = 0.9
weight decay = 5e−4
learning decay = 0.1 after every 20 epochs

The training loss and validation loss obtained for the original model at the end of 40th epoch was 0.51079 and 0.59293 respectively and the training accuracy and validation accuracy was 85.843 and 83.234 respectively. Similarly, the training loss and validation loss obtained for the modified model at the end of the 40th epoch was 0.10816 and 0.16873 respectively and the training accuracy and validation accuracy was 95.747 and 92.705 respectively.

The training charts obtained are as follows (**Figures 6** and 7).

Figure 5.
Sample images in the dataset to train the vehicle re-identification model.

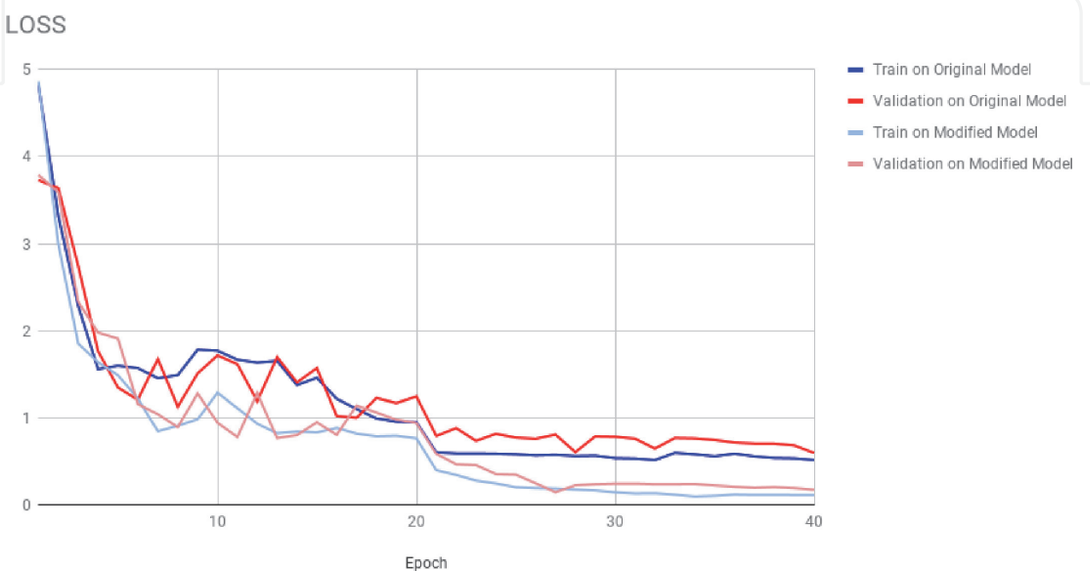

Figure 6.
Training loss and validation loss for both the original and modified re-identification networks.

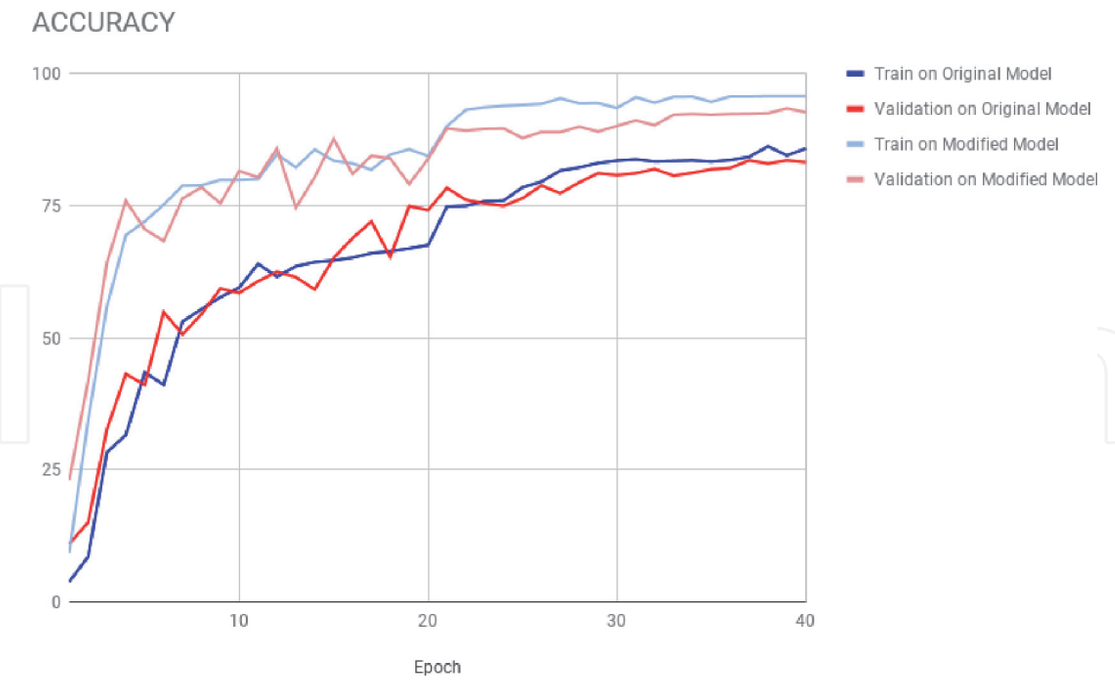

Figure 7.
Training accuracy and validation accuracy for both the original and modified re-identification networks.

4. Evaluation and results

The system was evaluated by feeding the live stream video from the CCTV camera as well as the pre-recorded videos. With the help of pre-recorded videos, both the original model and the modified model were tested. There was far less number of identity switches obtained for the modified model than the original model while running on the same video. The confidence score of YOLOv3 detector was set to 0.7. During occlusions as well as missed detections from the detector, tracking performance of the system with the modified re-identification model was far superior. The test videos can be viewed at the following hyperlinks. The first four videos are the outputs when tested with the original model and the last four are those when tested with the modified model.

The system runs at 15–20 fps on the hardware configuration mention in Section 3.2.

The system was evaluated based on detection accuracy and performance as well. MOT-16 [31], based on the CLEAR metrics [32] intuitively expresses the evaluation in two numbers- MOTP and MOTA.

Tracking system	MOTA	MOTP (%)	MT (%)	ML (%)	ID	FM	FP	FN
Deep SORT	61.4	79.1	32.8	18.2	781	2008	12852	56668
Proposed system	64.2	81.7	37.3	16.2	686	1985	12266	54089

MOTA: summary of general accuracy of tracking in relation to false positives, false negatives and identity switches; MOTP: summary of overall tracking precision in terms of bounding box overlap between ground-truth and reported location; MT: mostly tracked-percentage of ground-truth tracks with the same label for at least 80% of their lifetime; ML: mostly lost- percentage of ground-truth tracks tracked for a maximum of 20% of their life span; ID: identity switch-number of times the reported identity of a ground-truth track changes; FM: fragmentation-number of times a track is interrupted by a missing detection.

Table 1.
Evaluation table.

- *Multiple object tracking precision* (*MOTP*): expresses how well exact positions of persons are estimated.

- *Multiple object tracking accuracy* (*MOTA*): shows how many mistakes the tracker made in terms of misses, false positives, mismatches, failures to recover tracks.

The comparison of tracking with the original model with the proposed model in terms of MOTA and MOTP is as shown in **Table 1**. The accuracy and precision of tracking are increased in the proposed system and the number identity switches are significantly decreased.

5. Conclusion

The major objective of this work is to track the moving vehicles (cars) on the road. The various concepts of deep learning and computer vision have been utilized for this purpose. Track by detection framework was applied for real-time vehicle tracking. YOLOv3 object detection system was used to detect the vehicles and the concepts of Deep SORT algorithm was applied for tracking. By modifying the re-identification model of the original Deep SORT system and training the network on the vehicle dataset developed from scratch, the proposed system enhances the tracking performance by reducing the number of identity switches. With the use of more powerful hardware, the performance of the system can be enhanced.

The pros and cons of the proposed system are as follows.

5.1 Pros

- Fast-tracking due to the use of a very fast object detector.

- It can be used for real-time applications.

- High accuracy and reduced identity switches.

5.2 Cons

- The limitations include the necessity of both CPU and GPU. CPU is slow in calculations and is not suitable for real-time processing. Similarly, GPU cannot drive itself. It needs CPU to be controlled. Hence this system becomes a bit expensive in terms of cost and power consumption.

- The effectiveness of the algorithm is reduced if the bounding boxes from the object detector are too big because too much background information is captured in the features.

- The performance is slightly reduced during dark conditions.

Acknowledgements

This work is carried out at Internet of Things (IoT) Lab, Industrial Systems Engineering Department, Asian Institute of Technology, Thailand.

Videos

Videos available at: https://mts.intechopen.com/download/index/process/451/authkey/330918e03cab6552f87b111634173303

IntechOpen

Author details

Sandesh Shrestha
Asian Institute of Technology, Pathumthani, Thailand

*Address all correspondence to: sandeshshrestha45@gmail.com

References

[1] Bewley A, Ge Z, Ott L, Ramos F, Upcroft B. Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP). IEEE.; 2016. pp. 3464-3468

[2] Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP). IEEE.; 2017. pp. 3645-3649

[3] Jain AK, Zhong Y, Lakshmanan S. Object matching using deformable templates. IEEE Transactions on pattern analysis and machine intelligence. 1996; **18**(3):267-278

[4] Mundy JL. Object recognition in the geometric era: A retrospective. In: Toward Category-level Object Recognition. Berlin, Heidelberg: Springer; 2006. pp. 3-28

[5] Ponce J, Hebert M, Schmid C, Zisserman A, editors. Toward Category-Level Object Recognition. Vol. 4170. Berlin, Heidelberg: Springer; 2007

[6] Ojala T, Pietikinen M, Menp T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis & Machine Intelligence. 2002;**1**(7):971-987

[7] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005; Vol. 1; IEEE; 2005. pp. 886-893

[8] Lowe DG. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision. 2004;**60**(2):91-110

[9] Tuzel O, Porikli F, Meer P. Region covariance: A fast descriptor for detection and Classification. In: European Conference on Computer Vision. Berlin, Heidelberg: Springer; 2006. pp. 589-600

[10] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. 2012. pp. 1097-1105

[11] Khan G, Ghani MU, Siddiqi A, Seo S, Baik SW, Mehmood I, et al. Egocentric visual scene description based on human-object interaction and deep spatial relations among objects. Multimedia Tools and Applications. Vol. 77. Springer; 2018. pp. 1-22

[12] Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014. pp. 580-587

[13] Uijlings JR, Van De Sande KE, Gevers T, Smeulders AW. Selective search for object recognition. International Journal of Computer Vision. 2013;**104**(2):154-171

[14] He K, Zhang X, Ren S, Sun J. Spatial pyramid pooling in deep convolutional networks for visual recognition. In: European Conference on Computer Vision. Springer; 2014. pp. 346-361

[15] Girshick R. Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. 2015. pp. 1440-1448

[16] Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A. Learning deep features for discriminative localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016. pp. 2921-2929

[17] Ren S, He K, Girshick R, Sun J. Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems. 2015. pp. 91-99

[18] Redmon J, Divvala S, Girshick R, Farhadi A. 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

[19] Redmon J, Farhadi A. YOLO9000: Better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017. pp. 7263-7271

[20] Redmon J, Farhadi A. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767. 2018 Apr 8

[21] Dicle C, Camps OI, Sznaier M. The way they move: Tracking multiple targets with similar appearance. In: Proceedings of the IEEE International Conference on Computer Vision. 2013. pp. 2304-2311

[22] Yoon JH, Yang MH, Lim J, Yoon KJ. Bayesian multi-object tracking using motion context from multiple objects. In: 2015 IEEE Winter Conference on Applications of Computer Vision. IEEE.; 2015. pp. 33-40

[23] Kim C, Li F, Ciptadi A, Rehg JM. Multiple hypothesis tracking revisited. In: Proceedings of the IEEE International Conference on Computer Vision. 2015. pp. 4696-4704

[24] Bewley A, Ott L, Ramos F, Upcroft B. Alextrac: Affinity learning by exploring temporal reinforcement within association chains. In: 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE; 2016. pp. 2212-2218

[25] Reid D. An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control. 1979;**24**(6):843-854

[26] Fortmann T, Bar-Shalom Y, Scheffe M. Sonar tracking of multiple targets using joint probabilistic data association. IEEE Journal of Oceanic Engineering. 1983;**8**(3):173-184

[27] Rezatofighi AM, Zhang Z, Shi Q, Dick A, Reid I. Joint probabilistic data association revisited. In: Proceedings of the IEEE International Conference on Computer Vision. 2015. pp. 3047-3055

[28] Kayumbi G, Mazzeo PL, Spagnolo P, Taj M, Cavallaro A. Distributed visual sensing for virtual top-view trajectory generation in football videos. In: Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval. ACM; 2008. pp. 535-542

[29] Yang M, Jia Y. Temporal dynamic appearance modeling for online multi-person tracking. Computer Vision and Image Understanding. 2016;**153**:16-28

[30] Xiang Y, Alahi A, Savarese S. Learning to track: Online multi-object tracking by decision making. In: Proceedings of the IEEE International Conference on Computer Vision. 2015. pp. 4705-4713

[31] Milan A, Leal-Taix L, Reid I, Roth S, Schindler K. MOT16: A benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831. 2016

[32] Bernardin K, Stiefelhagen R. Evaluating multiple object tracking performance: The CLEAR MOT metrics. Journal on Image and Video Processing. 2008;**2008**:1