



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.


저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

Thesis for the Degree of Master of Engineering

Visual Multi-Object Tracking and Exploration based on Deep Learning

The logo of Pukyong National University is a circular seal. It features a stylized blue and white design in the center, possibly representing a compass or a flower. The outer ring of the seal contains the text "PUKYONG NATIONAL UNIVERSITY" in English at the top and "부경대학교" in Korean at the bottom.

by
Farkhodov Khurshedjon
Department of Artificial Intelligence Convergence
The Graduate School
Pukyong National University

February 19, 2021

Visual Multi-Object Tracking and Exploration based on Deep Learning

딥러닝 기반 시각적 다중 객체 추적 및 탐색

Advisor: Prof. Ki-Ryong Kwon

by

Farkhodov Khurshedjon

A thesis submitted in partial fulfillment of requirements for
the degree of

Master of Engineering

in Department of Artificial Intelligence Convergence
The Graduate School, Pukyong National University

February 2021

Visual Multi-Object Tracking and Exploration based on Deep Learning

A thesis

By

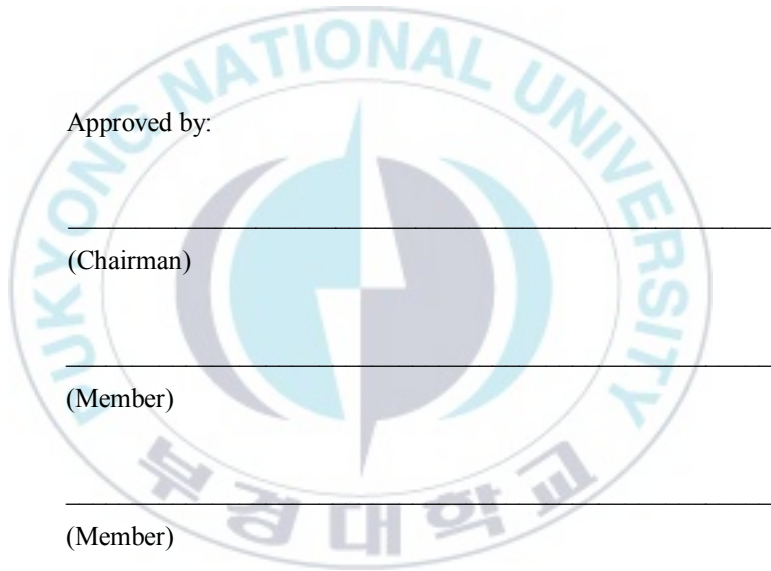
Farkhodov Khurshedjon

Approved by:

(Chairman)

(Member)

(Member)



February 19, 2021

Contents

I. Introduction.....	1
II. Related Works	6
2.1 Detection based object tracking.....	6
2.1.1 Faster RCNN object detection architecture.....	7
2.1.1.1 Convolutional layer.....	8
2.1.1.2 Region proposal networks	10
2.1.1.3 Classes and bounding boxes prediction.....	10
2.2 Data association-based object tracking.....	11
2.2.1 LSTM Network based object prediction.....	12
2.2.1.1 Sigmoid activation operation.....	14
2.2.1.2 Forget gate operation.....	15
2.2.1.3 Input gate operation	15
2.2.1.4 Cell state operation.....	15
2.2.1.5 Output gates operation	15
III. The Proposed Methods and Experimental Results.....	17
3.1 First experiment with proposed method: Object detection- based tracking.....	17
3.1.1 The drone-based dataset.....	17
3.1.2 Overview of the object detection task.....	18
3.1.3 The drone-based dataset training.....	20
3.1.4 Proposed object tracking method	24
3.1.5 Experimental Results and Discussion.....	28
3.2 The second experiment with proposed method: LSTM network with tracking association	33

3.2.1	Overview of the proposed multi-object tracking originality	33
3.2.2	Proposed LSTM network-based training	35
3.2.2.1	Proposed network model	36
3.2.2.2	Building LSTM network	37
3.2.2.3	Training configuration.....	38
3.2.3	Proposed LSTM tracking association.....	38
3.2.4	Experiment results and discussion.....	40
3.2.4.1	Training evaluation.	40
3.2.4.2	Tracking evaluation.....	41
3.2.4.3	Comparison results.....	42
IV.	Conclusion	45
	References.....	47

Figures

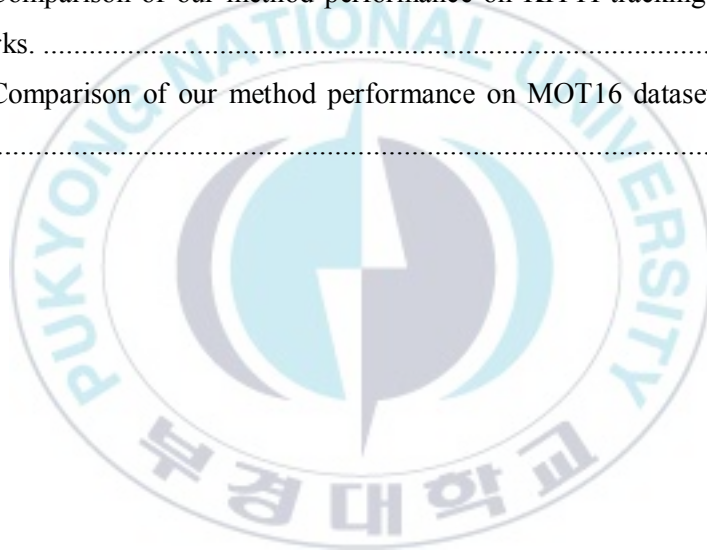
Fig. 1. The main architecture of the Faster RCNN.	7
Fig. 2. Common CNN structure for object classification.	8
Fig. 3. Calculation of feature map.	9
Fig. 4. Reducing the spatial size of the representation in an image by diminishing the number of parameters and computation in the network.	9
Fig. 5. Example of RPN structure.	10
Fig. 6. The Long Short-Term Memory (LSTM) cell can process data sequentially and keep its hidden state through time. 오류! 책갈피가 정의되어 있지 않습니다.	
Fig. 7. Basic steps for tracking an object.	17
Fig. 8. The drone-based dataset image examples.	18
Fig. 9. Tensor board losses results after 200K times training: a) in a box classifier / classification loss; b) in a box classifier / localization loss; c) in the RPN / localization loss; d) in the RPN / objectless loss	21
Fig. 10. Performance of the total losses along with clone loss	22
Fig. 11. Learning Rate and Global Step of training process	23
Fig. 12. Proposed tracking method structure	25
Fig. 13. Inference diagram of DNN module in OpenCV library.	27
Fig. 14. Inference diagram of CSRT tracker in OpenCV library.	27
Fig. 15. Object detection results of the pre-trained Faster RCNN object classifier: (a) image from internet sources taken by drone camera; (b, c) picture taken from drone video sequence; (d) image from VOT2018 dataset.	30
Fig. 16. Qualitative results of conventional (CSR-DCF, blue) and proposed (FRCNN_CSRT, green) trackers (0, 485, 981, ... - frames).	31
Fig. 17. The main steps of the proposed tracking method.	33
Fig. 18. Proposing LSTM Network model structure.	36

Fig. 19. Tracking association integration with the LSTM model.	39
Fig. 20. Qualitative results of the proposed method based on KITTI tracking dataset.	44



Tables

Table 1. Comparison results of proposed-neural network-based object tracking and conventional methods.	32
Table 2. Training evaluation of LSTM network model.	41
Table 3. LSTM network based multi object tracking evaluation on KITTI tracking dataset.	42
Table 4. Comparison of our method performance on KITTI-tracking dataset with recent works.	43
Table 5. Comparison of our method performance on MOT16 dataset with recent works.	43



딥러닝 기반 시각적 다중 객체 추적 및 탐색

Farkhodov Khurshedjon

부 경 대 학 교 대 학 원 인 공 지 능 융 합 학 과

요 약

현재 시각 물체 추적은 AI 와 로봇의 절차를 자율적으로 감시하는 데 필수적인 역할을 하는 제어 시스템의 새로운 기술적 발견과 과학적인 연구에 의해 대부분 관심을 끌고 있다. 일반적으로 시각적 객체 추적의 의미는 영상 시퀀스 프레임에서 움직이는 대상을 감지하는 과정으로, 물체의 초기 위치를 파악하여 추적 지점을 설정하고, 추가로 검출된 각 대상에 고유한 ID 를 부여하며, 비디오 시퀀스의 시간대에 프레임에서 프레임으로 이동하는 동안 추적한다. 본 연구에서는 시각적 객체 추적을 위해 시각적 추적 모델의 개발을 위한 보다 조정 가능한 방법을 발견하기 위한 두 가지 연구 포인트를 제시하였다. 첫 번째 연구에서는 우리가 제안했던 CSR-DC(Channel and Spatial Reliability of Discriminative Correlation) 객체 탐지 모델의 통합과 함께 흔히 사용되는 Faster RCNN (Region-based Convolutional Neural Network) 추적 필터 중 하나를 적용하여 최첨단 방법의 성능에 주목하였다.e 시각적 객체 추적의 가장 중요한 과제를 극복하기 위해 신경망 기반 추적 시스템을 구축하는 경우, 심층 학습 기반 객체 감지 및 CSRT(Channel and

Spatial Reliability Tracking)와 결합된 알고리즘. CSRT Tracker 자체를 사용하는 대신, 우리의 통합 구현 방법은 훨씬 더 나은 결과를 제시했다. 연구를 더 지속하기 위해, 우리는 시각적 객체 추적의 새로운 연구 방법론을 탐구했고, 다음 연구는 다중 객체 추적 분야에서 진행되었는데, 다중 객체 추적을 위한 추적 협회와 함께 새로운 모델 LSTM (Long-Short Term Memory) Network 를 제안했다. 다중 객체 추적은 단일 개체와 상당히 다른데, 우리는 추가적인 도전에 직면하게 될 것이다. 객체 추적에서 가장 보편적이고 분명한 작업은 폐색, 모션 흐림, 유사성, 중복성, 환경적 변화 등이며, 다중 객체 추적에 있는 과제와는 별개로 우리는 열거된 문제들에 대해 몇 가지 추가적인 도전이나 보다 강화된 형태를 가지고 있다. 그래서 LSTM 네트워크 기반 추적 연결 방법은 네트워크 구조를 구축함으로써 제안되어 왔는데, 추적과 관련된 것은 추적 가능한 물체 위치, 외관, 동작 세부 정보를 직접 학습할 수 있게 해주고, 추가로 LSTM 네트워크는 물체 특징 및 기타 정보를 학습함으로써 음미할 수 있는 능력을 갖게 된다.e 네트워크 셀에 대한 세부사항과 그 정보를 시퀀스의 시계열에서 대상의 다음 이동을 예측하기 위해 사용하며, 이는 초기 위치에 따라 미래 예측이 될 것이다. LSTM 네트워크와의 공동 객체 추적 성능에 따라 병렬로 학습하고 추적하기 위해 더 강력하고 채택할 수 있다. 제안된 두 가지 방법 모두 온라인 모드에서 작동할 수 있으며, 이는 제안의 효율성과 효과를 증가시킨다. 첫 번째 접근방식은 자체 데이터셋을 적용했고, 두 번째 접근방식은 오픈소스 데이터셋을 사용했다. 기존의 연구 방법 결과와 비교했을 때, 두 경우 모두 높은 시각적 및 자격적 성과와 통합된 딥러닝 기반 모듈에서 우리의 제안이 갖는 중요성.

Visual Multi-Object Tracking and Exploration based on Deep Learning

Farkhodov Khurshedjon

Department of Artificial Intelligence Convergence,
the Graduate School
Pukyong National University

Abstract

Currently, visual object tracking has been attracted by the newest technological findings and scientific researches of controlling systems, which has an essential role for autonomously monitoring procedures of AI and robotics. Generally, the meaning of the visual object tracking is a process of detecting a moving target from the video sequence frames, by taking the initial location of the objects to set a tracking point, additionally, giving a unique IDs to each detected target, and track them as they move around frame to frame in a time period of the video sequence. In this research, for visual object tracking, we have presented two research points to discover more adjustable method for development of visual tracking models. In the first study, the main attention focused on the stat-of-the-art method's performance, by applying one of the commonly used a Channel and Spatial Reliability of Discriminative Correlation (CSR-DC) tracking filter with the integration of a novel Faster RCNN (Region-based Convolutional Neural Network) object detection model, that we proposed the combined algorithm with deep learning-based object detection and CSRT (Channel and Spatial

Reliability Tracking), in case of creating neural network based tracking system, as well as, to overcome most crucial challenging task of the visual object tracking. Our integrated implementation method presented much better results, instead of using CSRT tracker itself. In order to continue our researches further, we have explored a new research methodology in visual object tracking, our next research study was conducted in the field of Multi-object tracking, which we have proposed a novel model LSTM (Long-Short Term Memory) Network with Tracking Association for Multi-Object Tracking. Multi-object tracking is quite different from single one, which we are going to face additional challenges. Most common and obvious tasks in object tracking are occlusion, motion blur, similarity, overlapping, environmental variations and so on, apart from those in multi-object tracking we have some other additional challenges or more intensified form of the listed problems. So, the LSTM network based tracking association method has been proposed by building a network structure, which is associated with tracking can give us to learn the trackable objects location directly, appearance, and motion details, additionally, the LSTM network has an ability, that by learning objects features and other information, it will save that details to network cells and uses that information for predicting, next movement of the targets in the time series of the sequences, which it would be a future prediction according to their initial position. According to the performance of the jointly object tracking with LSTM network is more powerful and adoptable to learn and tracking parallelly. Both proposed methods can work in online mode, which increases the efficiency and effectiveness of the proposals. In the first approach we have applied our own dataset, in the second one opensource dataset have been used. The significance our proposals, in both cases deep

learning-based module integrated with high visual and qualitative performance, compared with conventional and also with current research method results.



I. Introduction

Object tracking process is becoming one of the most commonly studied and daunting tasks in an image processing field. Generally, object tracking has been defined with many kinds of ideas, learning opinions, and statements, which describes the process briefly by its outcomes and consequences. Mostly, in many real-time applications requires remotely controlling unit for emergency situations which is more urgent to managing visually by using object tracking systems. In this study we are going to introduce about two research points, which clarifies the importance and advantages of object tracking in Computer Vision.

As a starting point of our research, in the first method, we begun by learning CNN-based structure for detecting and identifying objects from real time sequences. Also, the urgent part we should consider that during the real time object tracking, the trajectory of the objects in the actual time video frames may face problems, such as overlapping, occlusions, motion blur, changing appearance, illumination changes, cluttered background, environmental variations, and [1-3] etc. Mostly, that type of tricky and complex positions cannot be solved by applying simple tracking filters, algorithms, methods at all. Additionally, that type of methods cannot recover the objects and fail to retrack the targets in the following frames. Our proposal in such situation can overcome problems which we mentioned above and could perform better in object tracking operations. In the proposal creating object tracking process can be done by following several steps applying widely used algorithms and tips.

In this research we focused on firstly extracting the objects and track them through sequences. Furthermore, in real-time object tracking there are several situations to be considered while tracking in real time environment, for instance motion of the camera, range between trackable objects and camera, and so on. As a detection part of our model we applied Faster R-CNN based object detection model after exploring all kind of recent object detection models. As an example Viola-Jones

approach: the first efficient face detector [4], alternatively, there is an efficient detection technique: Histograms of Oriented Gradients [5], and from 2012 the deep learning methods, which is called “Convolutional Neural Networks” became the gold standard for image classification after Kriszhevsky's CNN's performance during ImageNet [6], While these results are impressive, image classification is far simpler than the complexity and diversity of true human visual understanding. In CNN based object detector there is image classification part, whereas an image with a single object as the focus and the task is to say what that image is. However, when we look around us, it's far more complex task to carry out. While detecting object from coming real-time frames it may appear with complicated sights including multiple overlapping objects, and different backgrounds, and not only classify these different objects but also identify their boundaries, differences, and relations to one another.

A better approach, R-CNN has been proposed [7] after CNN realized. R-CNN creates bounding boxes, or region proposals, using a process called Selective Search. Later, improvement of R-CNN has been recommended called Fast R-CNN by the same authors [8], which Fast R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Fast R-CNN employs several innovations to improve training and testing speed while also increasing detection accuracy. Furthermore, the authors of Fast R-CNN tried to find out more results and high detecting accuracy in every type of classes, and one more implementation of work Faster R-CNN suggested by R. Girshick [9], where further merge R-CNN and Fast R-CNN into a single network by sharing their convolutional features using the recently popular terminology of neural networks with “attention” mechanisms, the RPN component tells the unified network where to look [9].

There are several tracking filters, methods, algorithms have been implemented and utilized in variety of task that achieved noticeable outcomes. However, this actual algorithm of tracking pedestrians shows a low performance when faced with

problems like mentioned in a first paragraph of this chapter. In this paper, we propose an algorithm that to apply OpenCV-based CSRT tracker into person detection and tracking, which we combined with Faster R-CNN based object detector and obtained trained object detector model. With the great afford of deep learning-based object detection technique that we can easily avoid target misclassification and lost. As an object tracker we use OpenCV-based CSRT tracker that applies trained detection model which includes all object features, candidate regions, object bounds, objectless scores at each position, and exact labeled object. Our proposed algorithm associated with detection model and tracker at the same time. Moreover, the states of object are determined by detector and tracker that can use the information respectively while in tracking process.

In second study, Multi-Object Tracking research and its results is going to convey by proposing one of the Recurrent Neural Network based approach, called LSTM Network. Most of multi object tracking research outcomes has been shown that working with multiple objects the same time is more challenging and demanding task, when applying neural network-based approach. There are number of indeterminable and uncertain circumstances which we may face some difficulties to handle with RNN or CNN based tracking techniques. However, most of state-of-the-art approaches has been developed with the deep learning network [10-12]. Even though multi-object tracking systems are more complex and computationally overloaded for the real tools in case of working on accuracy measures and highly leveled functionalities of tracking systems. A number of attempts to figure out multi-object tracking most common challenges, such as, similar appearance, frequent occlusion, motion of several objects the same time, edge problems realized that adaption of deep learning approaches can face an additional charge that for analyzing unnecessary information from the video sequence and discrete set of detections. Moreover, here comes one more strategy of work is tracking by detection [13], whereas due to various challenges such as view point change, scales, density of

object distribution and occlusion, they have proposed a model for detection as a channel interdependencies by using "Squeeze-and-Excitation" (SE) blocks that adaptively recalibrates channel-wise feature responses which works with customized Deep SORT network for object detection along with deep association network for their tracking algorithm.

Our recent research in multi-object tracking has been achieved remarkable effectiveness along with high accuracy. The main goal in multi-object tracking is tracking or identifying multiple and different type of objects in video sequence. Currently, most of recent multi object tracking techniques that relayed on tracking by detection approach. However, those methods are not much efficient while their performances are fully depending on state-of-the-art detectors accuracy and proficiency of accomplishment while detecting in a different environment. These methods are mostly localization based on bounding box recovery, but that kind of methods can be helpful for finding an object on a flat image surface. In multi-object tracking process runs with a large number of parameters, where the deep detection models require huge amounts of inputs that can make the state-of-the-art approaches impasse or losing essential feature elements of object while tracking objects across in video frames. Our proposing model which we are going to describe, is contains two main part that a LSTM Network and tracking association which gives us a fully tracking results. The network we have created is formed with LSTM cell-based layers that to explore the training set of image sequences. Main idea of proposed method is to focus on the object's appearance and its location, by estimating the initial location of objects in starting point and saving it into network cells along with classified features of the sequence frames. Moreover, the whole process of learning and tracking steps are separately divided, initial process is to train our network with training dataset in offline mode and in a second step tracking multi-objects with the integration of tracking association. Tracking process goes online across a sequential video frames tracked objects with identical numbers. Our LSTM network is

applicable to get entire sequences of data to analyze and finding tracking object features and remembering values of bounding boxes over arbitrary time intervals into cells than will be controlled by three gates that the flow of coming information into and out of the cell.

Nowadays, the importance of multi object tracking is more urgent task than ever before, there are some fields that requires controlling and tracking identities of several type of objects. Generally, in such methods of tracking requires to use common and complex datasets in order to analyze problem in-depth and multiple cases. We have used two open source datasets for training and testing our observation in detail, these are KITTI [29] and MOT2016[30]. These datasets are quite common and eligible to test our approach, allows to observe our strategy with common real-time tracking performance. The results of our experiment bared that in multi object tracking identifying every object in frames and assigning them as identical item with their appearance is more complex to get accurate outputs. However, the performance of our tracking system was more accurate and high-performance results with visually qualitative outputs.

In this research study, both presented methods performed good results with high percentage, as well as, the solutions for eliminating given challenges are more accurate and uses simple steps to overcome the uncertain and tricky conditions of the object tracking process. Evidently, the given methods are much more efficient and adjustable to combine with other real applications.

II. Related Works

2.1 Detection based object tracking

Generally, Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain classis (such as humans, buildings, or cars) in digital images and videos [13]. Well-researched domains of object detection include face detection and pedestrian detection as well.

Object detection has applications in many areas of computer vision, including image retrieval and video surveillance. Detecting objects from image whether an example of a semantic object is in a 2D image or in a real-time video sequence, one could create a classification model trained with feature extracted from positive and negative classes and use the trained model to classify a given image [14]. Variations in position, color, lighting conditions, size, etc., greatly affect the performance of the model. Object detection and tracking processes should be fine-tuned, that depending on what kind of problem is going to be solved. This is usually determined by the characteristics of the target. For instance, detecting vehicles require different parameters tuning than detecting pedestrians, animals, or faces, and so on. Due to the variations of the target, the process of detection differs with internal parameters, such as environmental conditions, target size, change in appearance, etc.

This feature-based technique exploits a notable differentiation character of the objects that taken from 2D pixel information in an image [15]. While using feature points of 2D images, such as color, intensity, background information it is easy way to identify object from frames if it will not change the appearance, position, and size as well. However, this 2D pixel information of images usually cannot give us proper and exact parameters of observed objects. Deep learning approaches for object detection has achieved state-of-the-art performance and used several object detection applications. This method gives better results rather than other detection techniques

and can resolve the problems mentioned in the introduction part. However, the indicated way of detection has required specific hardware and software systems for designing object detection model. There are certain high-level APIs (Application Programming Interface) like TensorFlow, Keras, and JSON that a set of functions and procedures allowing the creation of applications, that can access the features or data of an operating system, application, or other service for building a deep learning-based object detection model. According to the given issue and our goal in all respects for building detection model we have used TensorFlow API model and Faster RCNN an object detector architecture presented by R. Girshick, and his colleagues in 2015 [9].

2.1.1 Faster RCNN object detection architecture.

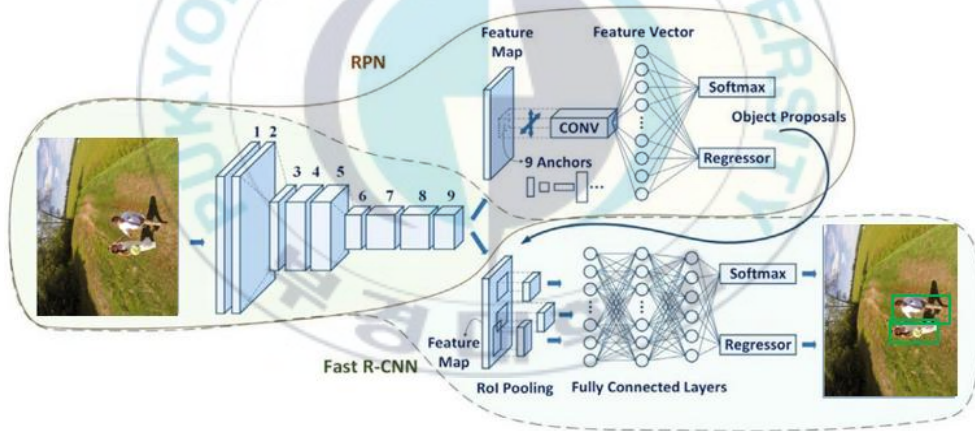


Figure 1. The main architecture of the Faster RCNN.

Faster RCNN is becoming one of the most used and popular an object detection architecture presented by R. Girshick, Sh. Ren, K. He and J. Sun in 2015 that uses Convolutional Neural Network like other famous detectors, such as YOLO (You Look Only Once), SSD (Single Shot Detector) and so on. In general, Faster RCNN composed from three main part at all that can be

managed building object detection model process. They are: a) convolution layers; b) region proposal network; c) classes and bounding boxes prediction, that shown in Figure 1 above [16].

2.1.1.1 Convolutional layer

These layers can help to extract the appropriate features of the images from dataset via training filters that will learn through training shapes and colors that exist in training images. Our target to detect and track is person or we can say pedestrian as well. Learning rate of object detection depend on the quantity of images and structural layer combination of the CNN. As an example of learning features process, we can assimilate filtration of the ready coffee. As a convolution layers to coffee filter that coffee filter does not let the coffee powder pass to the cup, so in this case also convolutional layers learn the object features and don't let anything else pass without learning, only the targeted object. In a detail, we can improve this activity by making labeling images manually as well, and as a result the ready coffee liquid will be last feature map of the CNN. Generally convolutional network composed of Convolution layers, pooling layers, activation layers and last component of the layers called fully connected layer or another extended thing that will be applied for an appropriate task like classification or detection. There is common structure of the CNN structure shown Figure 2 [16]:

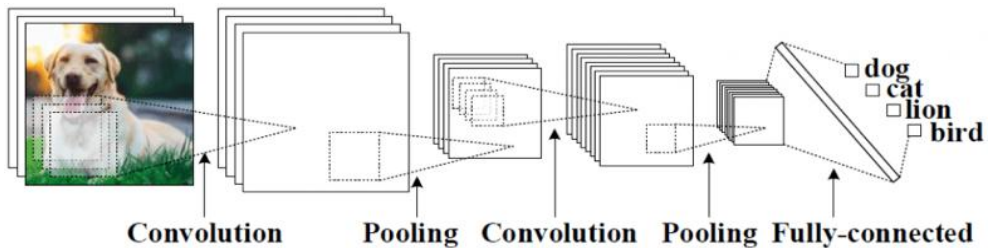


Figure 2. Common CNN structure for object classification.

By sliding filter along throughout input image, we compute convolution and result will come out as a two-dimensional matrix called feature map within in anchors that shown in Figure 3:

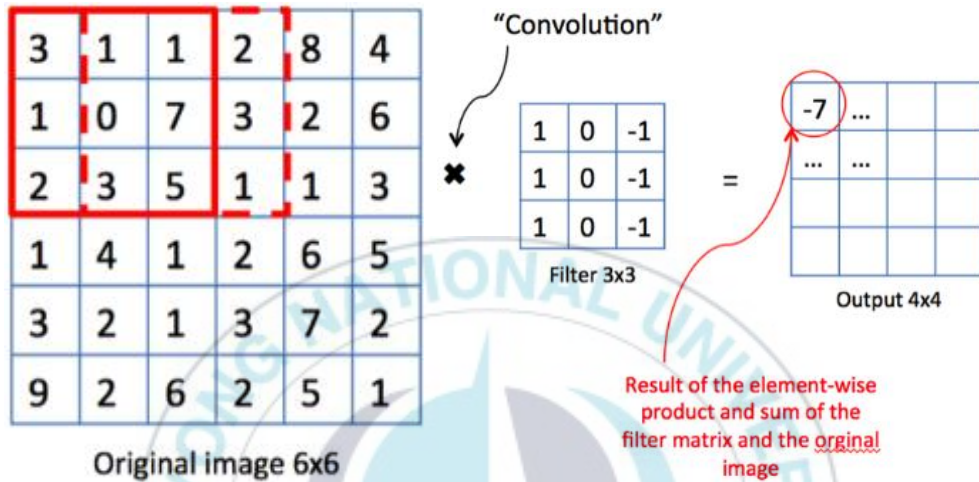


Figure 3. Calculation of feature map.

After computing feature map comes pooling layer which consist of decreasing quantity of features that eliminates pixels with low values from feature map shown in Figure 4:

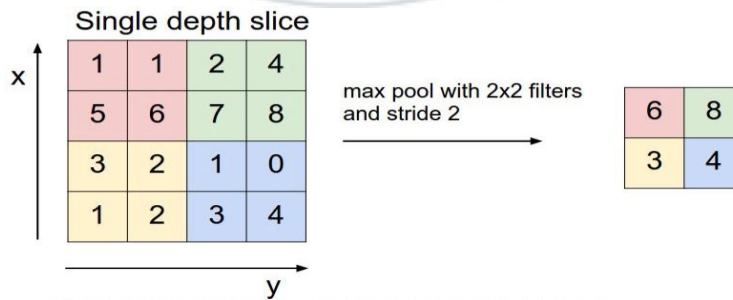


Figure 4. Reducing the spatial size of the representation in an image by diminishing the number of parameters and computation in the network.

2.1.1.2 Region proposal networks

RPN is a small neural network to generate proposals for the region where the object located, that a small sliding anchor box over a convolutional feature map and predict whether there is an object or not and also predict the bounding box of those objects shown in Figure 5 [17, 9]:

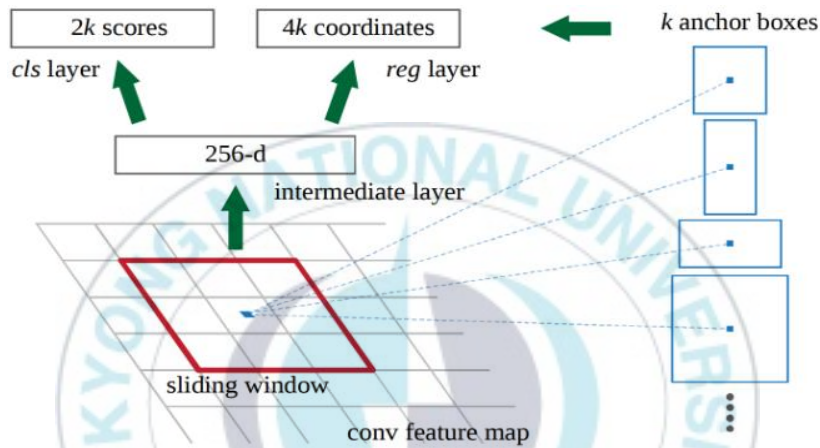


Figure 5. Example of RPN structure.

RPN has a classifier that determines the probability of proposal having the target object, while regression regresses the coordinates of the proposals. In Figure 5 shown that k anchors boxes applied for each location, which means anchors are translation invariant that uses the same ones at every location. Regression gives offsets from anchor boxes that each (regressed) anchor shows an object after classification gives the probability.

2.1.1.3 Classes and bounding boxes prediction.

This part is final step of object detection that we use fully connected neural network layer, and as an input the regions are proposed by the RPN and predicts object classes (Classification), Bounding boxes (Regression) as well.

2.2 Data association-based object tracking

Currently, the tracking-by-detection method has been successful as one of the most common techniques in the field of Computer Vision by the result of contemporary research strategies of object detection [18], methods, and its integration with other tracking algorithms [19]. Additionally, to overcome that kind of problems in recent multi-object tracking approaches started improving their technique with data association [20], to provide a tracking system with detected object information in terms of continuously tracking. As we mentioned before, an essential part of multi-object tracking is the memory unit, where we are going to detect and track several objects that similar to each other. In visual multi-object tracking identifying each object as one, an identical object is important, and similarity between multi objects has a considerable role in case of identifying them separately [21]. Exploring the entire image as a whole grid instead of computing exactly object located regions of interest will decrease the accuracy of detection and may lead to overwhelming of object features and misclassification.

In the last few years, one more technique has fascinated several field researchers working on sequential data modeling with different outputs. It's called the LSTM network that was proposed in 1997 by Sepp Hoch Reiter and Jurgen Schmid Huber [22]. The root concept in LSTM has been introduced by solving Constant Error Carousel (CEC) units, that deals with the vanishing gradient problem. LSTM is one type of an artificial Recurrent Neural Network's class architecture that contains basic elements of RNN and widely used in the field of deep learning. Because of its capability and eligibility, it can work not only process of learning single data points (such as images), alternatively, it can be applied to the entire sequential data, such as speech or video processing. For example, there are several LSTM based works have been purposed in a different field of sequential data processing, such as handwriting recognition, speech recognition, and anomaly detection in network traffic or IDS (Intrusion Detection Systems) [23]. Most of the regularly used LSTM

architecture is composed of a cell, an input gate, an output gate and a forget gate, where the entire process of regulation flow of information into and out of the cell will be conducted by over arbitrary time intervals remembering learned values of the sequential data. There are some well-known and popular LSTM networks that well-suited for classifying, processing, and making predictions based on time series data. Because of the indefinite sequential data time series or unknown duration between important events in a time series, the LSTM network capable of taking vital part of the sequence data and dealing with the vanishing gradient problem that can be encountered when training traditional RNNs. Common architecture of the LSTM presented below in Figure 6 [22], and a detailed description of the architecture comes following units.

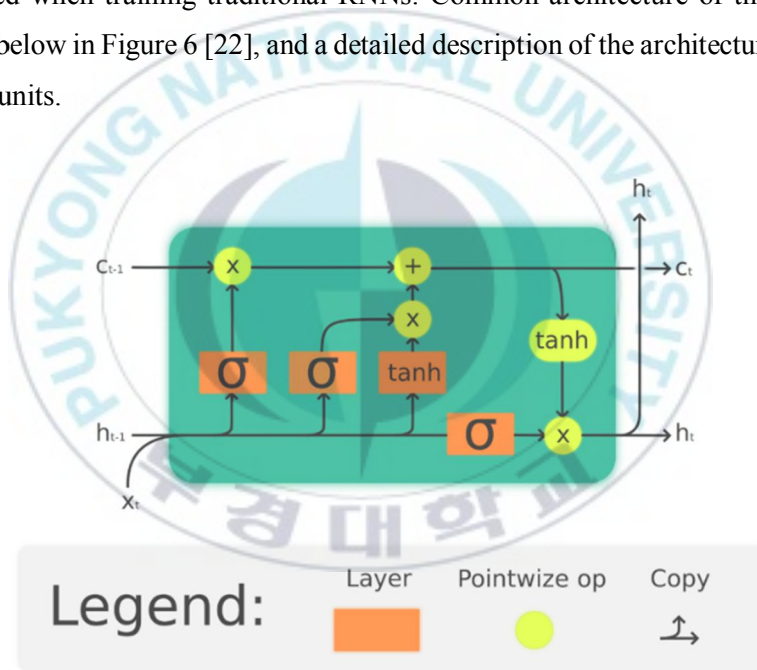


Figure 6. The Long Short-Term Memory (LSTM) cell can process data sequentially and keep its hidden state through time.

2.2.1 LSTM Network based object prediction.

The core concept of the LSTM network looks similar in a controlling flow chart as a recurrent neural network that the data passing process goes on information as it

propagates forward. Theoretically, the cell states can carry or keep temporarily relevant details across all the processing of the sequences. From the name of the method, even detailed particulars from initial time steps can make its way to later time steps, reducing the effects of short-term memory. As we mentioned before, the LSTM cells have gates, while the cell state goes on its way, features get inserted or erased it to the cell state via gates. The LSTM network gates perform a different task while training, also they decide which information is allowed on the cell state, additionally they can learn which features are necessary to keep or forget it during the training process. Network operations have their tasks to do in every step, inside of the LSTM cell there are operations: sigmoid, tanh, pointwise multiplication, pointwise addition, and vector concatenation. These operations control learned feature information of the network while predicting and tracking process as well.

The mathematical equations of the LSTM for the forward pass of the cell state unit with a forget gate are given below, which describes the calculation of the operations mathematically [22]:

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \quad (1)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$

$$\hat{c}_t = \sigma_g(W_c x_t + U_c h_{t-1} + b_c) \quad (4)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{c}_t \quad (5)$$

$$h_t = o_t \circ \sigma_h(c_t) \quad (6)$$

where the initial values are $c_0 = 0$ and $h_0 = 0$ and the operator \circ denotes the Hadamard product (element-wise product). The subscript t indexes the time steps.

The description of the variables:

- $x_t \in \mathbb{R}^d$: input vector to the LSTM unit
- $f_t \in \mathbb{R}^h$: forget gate's activation vector

- $i_t \in \mathbb{R}^h$: input/update gate's activation vector
- $o_t \in \mathbb{R}^h$: output gate's activation vector
- $h_t \in \mathbb{R}^h$: hidden state vector also known as output vector of the LSTM unit
- $\hat{c}_t \in \mathbb{R}^h$: cell input activation vector
- $c_t \in \mathbb{R}^h$: cell state vector
- $W \in \mathbb{R}^{h \times d}, U \in \mathbb{R}^{h \times h}$ and $b \in \mathbb{R}^h$: weight matrices and bias vector parameters which need to be learned during training

where the superscripts and refer to the number of input features and number of hidden units, respectively.

Activation functions:

- σ_g : sigmoid function.
- σ_c : hyperbolic tangent function.
- σ_h : hyperbolic tangent function or as the peephole LSTM paper suggests $\sigma_h(x) = x$.

2.2.1.1 Sigmoid activation operation

The sigmoid function has a great role in learning details and controlling by cell state unit gates. Generally, a sigmoid function performs the same as the tanh activation, but a sigmoid activation takes between 0 and 1 values for squishing, instead of values between -1 and 1. That form may help to update or forgetting the process of the cell state unit's gates. Because any taken value number multiplies 0 the outcome also becomes 0, which means causing values gives results to disappearing or forgotten. If its opposite, when any number multiplied by 1 is the same value stays the same or kept. Hence, the network decides which data is will be taken or forgotten while learning.

2.2.1.2 Forget gate operation

Firstly, the forget gate is an essential unit of the LSTM network that decides what information should be thrown away or kept. Initially, the information from the previously hidden state h_{t-1} and the current input x_t is passed through the sigmoid function σ_g . As an output the values come out between 0 and 1, if that values closer to 0 is going to forget f_t (1), else 1 means to keep it as a feature.

2.2.1.3 Input gate operation

For updating the cell state-input gate will be used, previously, the hidden state and current input together pass through a sigmoid activation σ_g , and the same input goes to tanh function at the same time that pointwise would be multiplied as well as the equation (2) will decide which input values will be updated by passing values between 0 and 1, if the value is 0 means not important, or 1 means important. When the input passes through the tanh function, it will be squished values between -1 and 1, to regulate the network.

2.2.1.4 Cell state operation

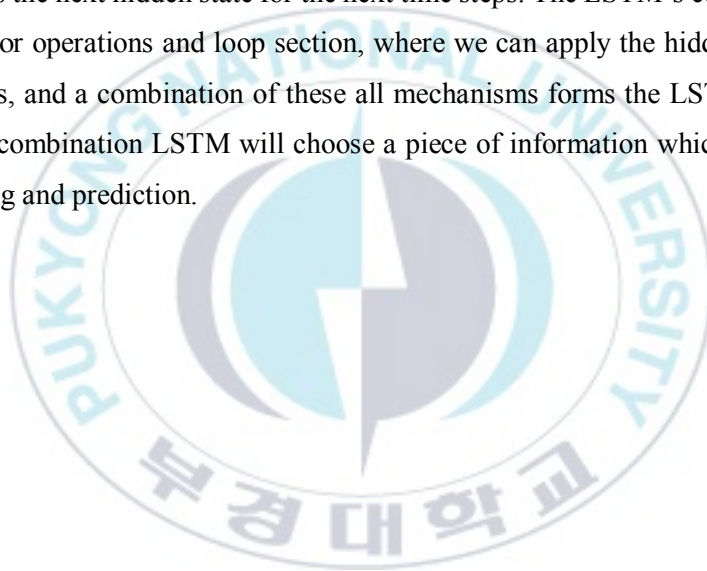
Now all passed forget and input gate information goes to the calculation of the cell state, initially, the cell state gets pointwise c_{t-1} multiplied by the forget activation f_t vector (4), that stage has a possibility of dropping values from the cell state if it gets multiplied by values neat to 0. After taking the output from the input gate will be pointwise addition that updates the cell state to new values that the neural network finds relevant to keep for prediction (5), which gives us our new cell state.

2.2.1.5 Output gates operation

In the final stage, the output gate determines which next hidden state h_t should be (6), that, the hidden state contains information on previous inputs, also used for predictions. Firstly, the previous hidden state and the current input goes through into a sigmoid activation (3), secondly, a newly modified cell state will be passed from a

tanh function. Then two output values would be multiplied to decide what information the hidden state should carry, that output is the hidden state. The new cell state and the new hidden are then carried over to the next time step learning process.

The rest time steps learning and prediction process follows as described above, where the forget gate decides that what learned information is relevant to keep from the initial time steps. Moreover, for updating information the input gate will decide to add necessary learned features from the current step. Lastly, the output gate determines the next hidden state for the next time steps. The LSTM's control flow is a few tensor operations and loop section, where we can apply the hidden states for predictions, and a combination of these all mechanisms forms the LSTM network, from that combination LSTM will choose a piece of information which is relevant for learning and prediction.



III. The Proposed Methods and Experimental Results

3.1 First experiment with proposed method: Object detection-based tracking

Tracking process is quite demanding task in case of using in a real-time controlling system. In general, the problem we are going to find a solution is starts with learning what object is going to track and collecting dataset related to our target. Suggested Object detection method in this proposed object tracking is one of the most useful and popular detection structures among detection approaches that gives good results faster and high accuracy. Tracking algorithm has been implemented to correspond each other while tracking process and get required information from trained object detection model. Figure 7 summarizes general principal steps used in our first method below:

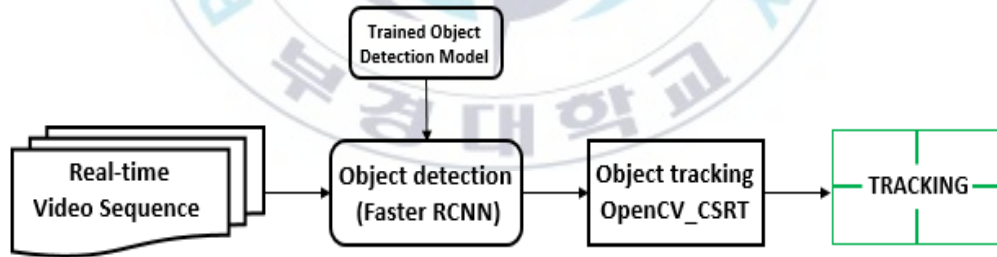


Figure 7. Basic steps for tracking an object.

3.1.1 The drone-based dataset

The dataset has been collected through different kind of internet resources, like post blocks, media sites, google resources, and so on. All pictures on dataset taken by drone camera in a different positions and backgrounds. After collecting all needed

images, it has been labeled into two classes: person and car. Those labeled images stored into required direction to create a label map for identifying object location, bounding boxes, and object features from dataset. A difference between our dataset from other open source datasets are: in our dataset only two type of objects have been labeled and created label map after detection for identifying object's ID and name as well. Our dataset contains 600 labeled images, 400 for training and 200 for testing. All images have been labeled manually and created .csv files for training and test labeled images, which contains coordinates of bounding rectangles for every object in every image that stored image folder as a ready dataset files to run for training process. Here are some images shown in Figure 8 from our drone-based dataset.

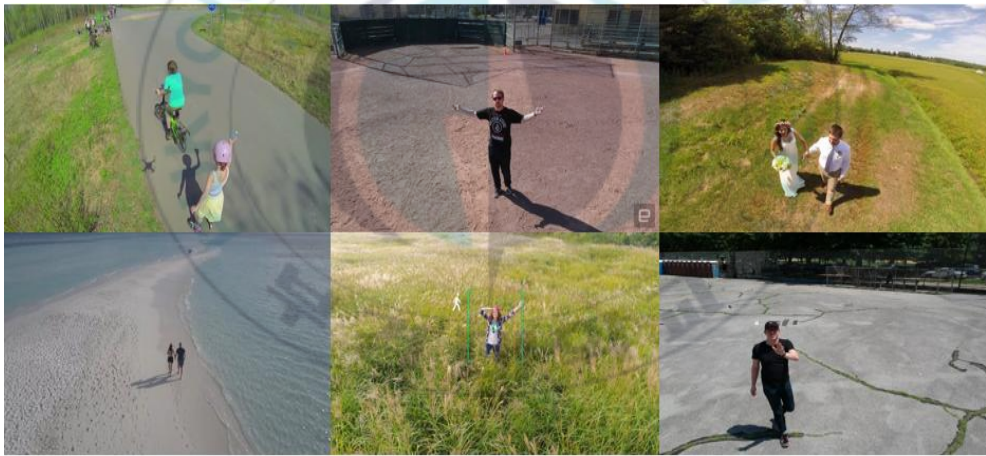


Figure 8. The drone-based dataset image examples.

3.1.2 Overview of the object detection task

Generally, Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain classis (such as humans, buildings, or cars) in digital images and videos [13].

Well-researched domains of object detection include face detection and pedestrian detection too.

Object detection has applications in many areas of computer vision, including image retrieval and video surveillance. Detecting objects from image whether an example of a semantic object is in a 2D image or in a real-time video sequence, one could create a classification model trained with feature extracted from positive and negative classes and use the trained model to classify a given image [14]. Variations in position, color, lighting conditions, size, etc., greatly affect the performance of the model. Object detection and tracking processes should be fine-tuned, that depending on what kind of problem is going to be solved. This is usually determined by the characteristics of the target. For instance, detecting vehicles require different parameters tuning than detecting pedestrians, animals, or faces, and so on. Due to the variations of the target, the process of detection differs with internal parameters, such as environmental conditions, target size, change in appearance, etc.

This feature-based technique exploits a notable differentiation character of the objects that taken from 2D pixel information in an image [15]. While using feature points of 2D images, such as color, intensity, background information it is easy way to identify object from frames if it will not change the appearance, position, and size as well. However, this 2D pixel information of images usually cannot give us proper and exact parameters of observed objects. Deep learning approaches for object detection has achieved state-of-the-art performance and used several object detection applications. This method gives better results rather than other detection techniques and can resolve the problems mentioned in the introduction part. However, the indicated way of detection has required specific hardware and software systems for designing object detection model. There are certain high-level APIs (Application Programming Interface) like TensorFlow, Keras, and JSON that a set of functions and procedures allowing the creation of applications, that can access the features or data of an operating system, application, or other service for building a deep

learning-based object detection model. According to the given issue and our goal in all respects for building detection model we have used TensorFlow API model and Faster RCNN an object detector architecture presented by R. Girshick, and his colleagues in 2015 [9].

3.1.3 The drone-based dataset training

Most open source datasets have been applied for classifying objects and identifying them as well. There are several datasets which contains more than 1000 classes and each class has training and testing images, that most common datasets: ImageNet, MNIST, LSUN, MS COCO – Common Objects in Context, Youtube-8M, Yelp Reviews, CIFAR-10, Open Image Dataset and so on. We collected our own dataset which has 600 images at all, of these 400 for training and 200 for testing images. As we mentioned before, we labeled all images manually and created label map for classifying them into two classes: pedestrian and vehicle. There is no exact number of labeled person or car classes, that most of images have more than one object and car or person class. These images have been stored into needed direction and created train_label and test_label .csv files which holds coordinates of the bounding rectangles for every object in every image. After that by using created .csv files training and testing tfrecords have been generated where we can use to read data efficiently and it can be helpful to serialize our data and store it in a set of files that can each be read linearly.

Now we need to configure object detection pipeline that defines which models and what parameters will be used for training and points through the training images and data. Our training model is called Faster RCNN Inception V2 which we will train our dataset by editing some parameters and passing it to training folder. While editing this model we have to change number of classes which we want to classify and detect. Moreover, there is some file paths like checkpoint, tfrecords, label map has been changed, also number of training and testing images quantity

changed to exact number in images folder. After setting up all parameters and needed configurations our selected model ready to training process. If everything has been set up correctly tensorflow will initialize the training and initialization can take 30 seconds before the actual training begins.

Losses

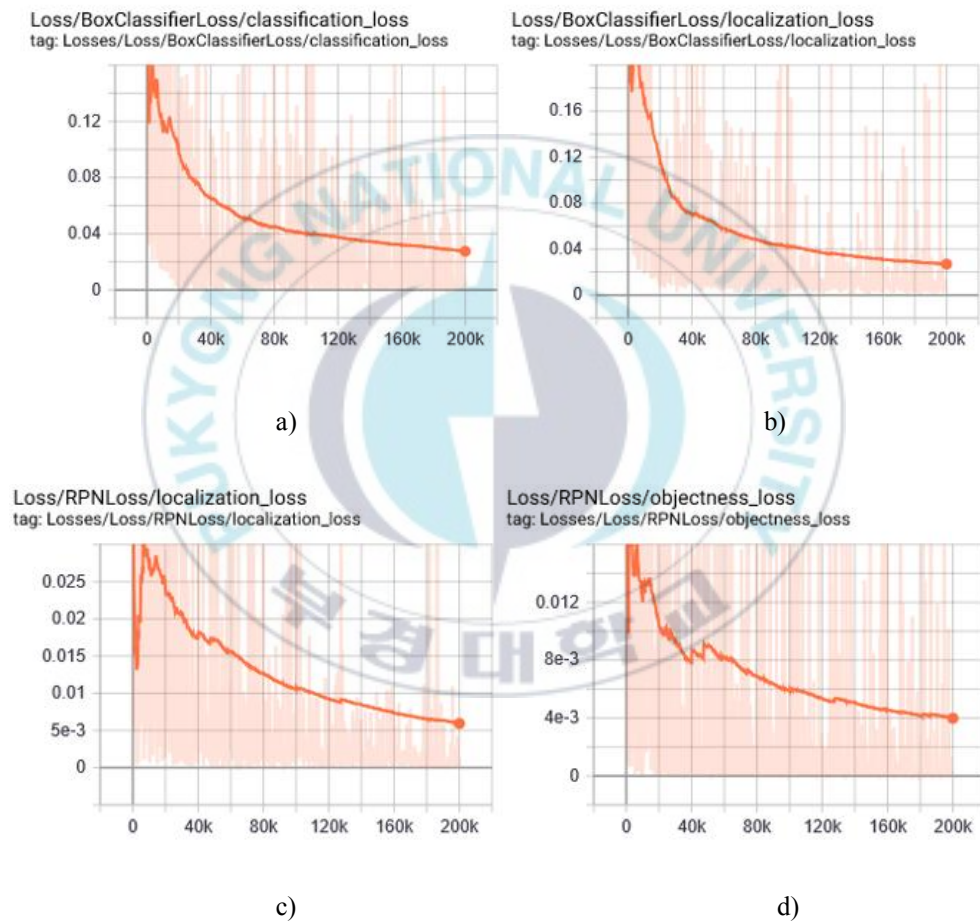


Figure 9. Tensor board losses results after 200K times training: a) in a box classifier / classification loss; b) in a box classifier / localization loss; c) in the RPN / localization loss; d) in the RPN / objectless loss.

Training process speed and time depends on what kind of CPU and GPU system we have, if our OS has last version of GPU hardware system that means we can get results faster than using CPU system, while training time we recommend not doing anything on your computer that requires lots of processing power, because training classifier uses 100% of your CPU and GPU. The training process begins with stepping through training batches and reporting the loss at each step and the law sloth starts off high and then gets lower and lower as the object detection classifier trains. Training classifier should train until the loss is consistently below 0.05 or so that the law starts to plateau out. We can view the process of training job by using tensor board to set up it on our localhost and view through a web browser.

In figure 9 has shown the result of training process after 200k times at all, where we can see the classifier has reached lower than 0.05 in a first (a) graph that means our chosen model for training has showed a good performance. A total loss graph estimates that while learning training dataset images it can loss or misidentify objects by their features, shapes and other parameters in one average graph performance. The total loss of training process performance together with clone loss given in Figure 10 below:

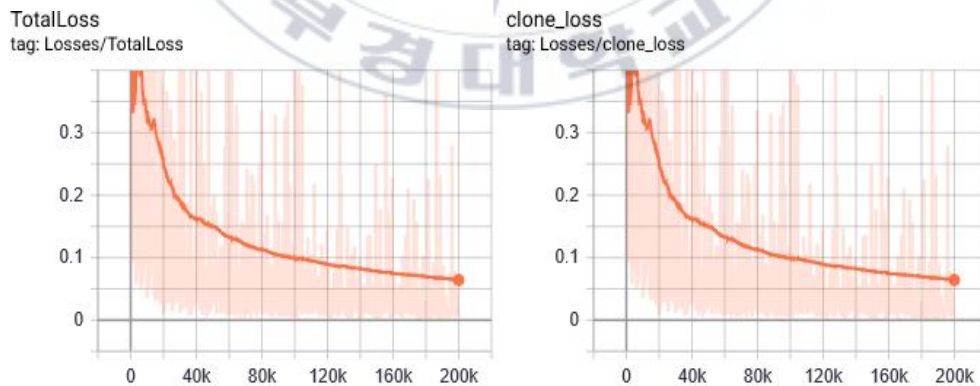


Figure 10. Performance of the total losses along with clone loss.

One more essential part of graphical result of training process are learning rate of trained and tested dataset and step of learning, which shows how quickly the model is adapted to the problem in the range of -1 and 1, but exact learning rate is $2e-4$ shown in Figure 10. Specifically, the learning rate is a configurable hyper parameter used in the training of neural networks that has a small positive value, often in the range between 0.0 and 1.0 [24]. Next graph named Global step is shows that time where the Faster RCNN object detection model took time for learning dataset images one by one. In this graph we can see average step for learning one step took 0.28 second, also it depends on how many images in dataset as well. Tensorflow saves training checkpoints every five minutes or so and stores them into the training folder, where the checkpoint of highest step count will be used to generate a classifier. Finally, after training 200k times finished in a training folder there are some generated training result files like checkpoint, model. ckpt. meta, model. ckpt. index, pipeline.config, and so on, which we can use to export inference frozen graph as an object detection model or object classifier.

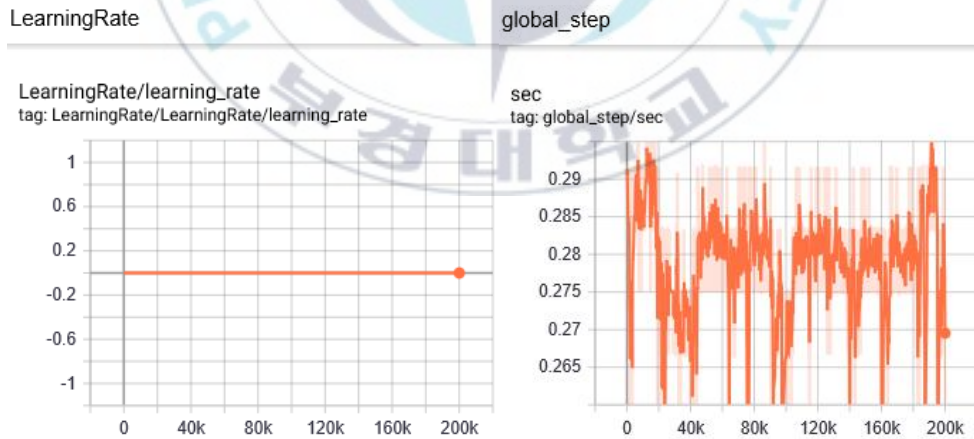


Figure 11. Learning Rate and Global Step of training process.

The last step is to generate a frozen inference graph by using the step count of the highest model checkpoint in the training folder that in our case it's two hundred thousand (200k). By typing needed command line code, we will create a frozen inference graph.pb file into the given output path. This .pb files contains the object detection classifier that will be used as an object detection model while object tracking process.

3.1.4 Proposed object tracking method

Object Tracking is one of the most challenging tasks in Computer Vision and it has an important domain in this sphere. It involves the process of tracking an object which could be a person, car or any type of movable objects across a series of frames. Our proposed tracking system related to tracking people which we would start with all possible detection in a frame by using Faster RCNN object detection model. Even in subsequent frames person will move away from the frame our detection model will identify the object with any position of it across a series of frames. The basis of our tracking method taken from the Discriminative Correlation Filter with Channel and Spatial Reliability (CSR-DCF) [25] tracking algorithm. Moreover, this algorithm has been implemented in a C++ and integrated into Open CV library as a DNN (Deep Neural Networks) module [26]. We proposed a tracking system that integration of Faster RCNN object detection as an object detector and OpenCV_CSRT_tracker as a tracking algorithm for tracking method that shown in Figure 12.

We have integrated CNN based object detector with OpenCV tracker where we will apply Faster RCNN based object detector model to support tracking algorithm with stable object identifier from coming real time frames. As an Object detection model have been used output of the Faster RCNN object classifier file that will support to identify objects from coming frames and gives output predictions to the CSRT tracker.

Our proposed integrated algorithm from taking real-time frames till tracking process pseudocode given in Algorithm 1.

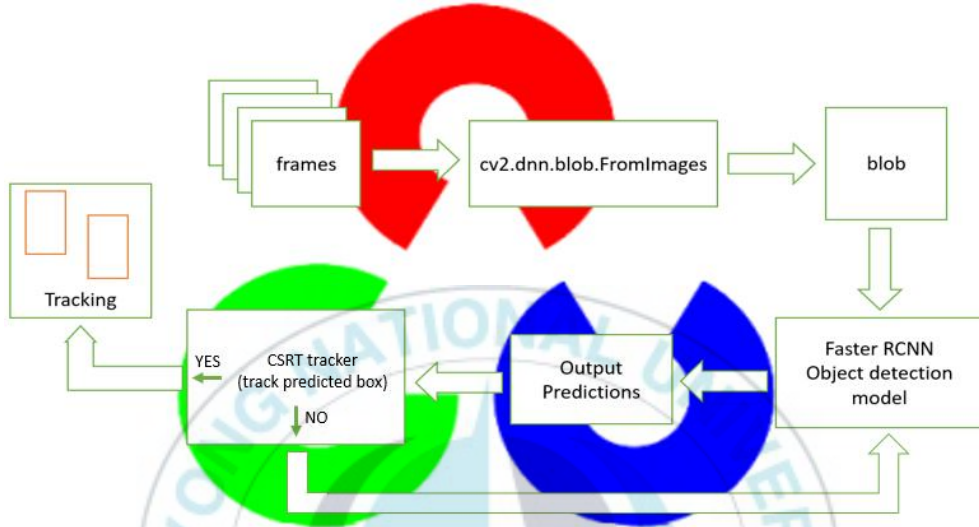


Figure 12. Proposed tracking method structure.

To support object detection model with OpenCV tracker has been used OpenCV dnn module in the library that implements forward pass (inferencing) with deep networks, pre-trained object classifier that used one of the most popular deep learning framework TensorFlow. DNN module contains two main functions that can be used for preprocessing images and preparing them for classification via pre-trained deep learning model.

Algorithm 1: The FRCNN_CSRT integrated tracking algorithm

1. Loop over frames of the sequence, and store corresponding information from each frame;

2. Resize a frame and compute the absolute difference between the current frame and first frame;
 3. Loop over the contours identified and compute the bounding box for the contour, draw it on the frame, (x, y, w, h), and use tensorflow object detection model files to detect object;
 4. Extract the index of the class label from the 'detections', then compute the (x, y)-coordinates of the bounding box for the object;
 5. Draw the prediction on the frame and start tracker with differing initial frame and initial bounding box;
 6. Check to see if we are currently tracking an object, if so, ignore other boxes this code is relevant if we want to identify particular persons and grab the new bb coordinates of the object;
 7. Check to see if the tracking was a success and update the FPS counter;
 8. loop over the info tuples and draw them on our frame and draw the text and timestamp on the frame;
 9. Show the frame and record if the user presses a key.
-
-

Structure of the dnn module allows to create and manipulate comprehensive artificial neural networks [26]. Neural network is presented as directed acyclic graph (DAG), where vertices are Layer instances, and edges specify relationships between layers inputs and outputs. In dnn library in each network layer given unique integer id and unique string name as well inside of the network. Moreover, layer ID can store either layer name or layer ID, also this class supports reference counting of its instances, i. e. copies point to the same instance. Inference diagram for cv::dnn::Net shown in Figure 13 below:

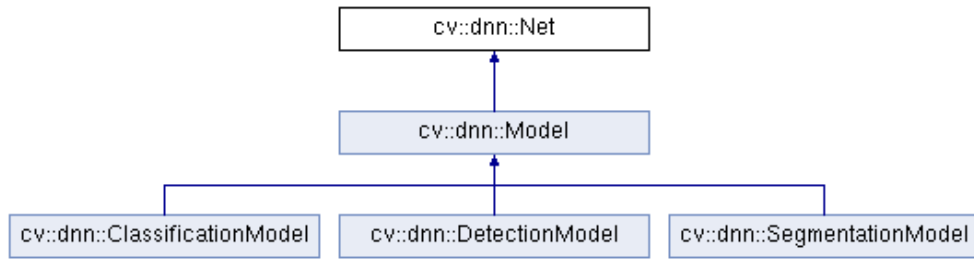


Figure 13. Inference diagram of DNN module in OpenCV library.

The given diagram shows main structure of the dnn net where we have been applied Classification Model and Detection Model only to track objects wherewith our pre-trained object classifier. DNN module supports all type of CNN (33 different) layers. Our object detection method Faster RCNN also built on with the help of some dnn layers where OpenCV dnn module can support easily. Detection model of OpenCV represents high-level API for object detection networks, also allows to set parameters for preprocessing input image. Furthermore, it creates net from with trained weights and config, sets preprocessing input, runs forward pass and return result detections, also supports SSD, Faster RCNN, and YOLO topologies. Also, Classification part the same with detection model, it also uses trained weights and config files, sets preprocessing input, runs forward pass and return top - 1 prediction.

OpenCV CSRT object tracker implementation is based on Discriminative Correlation Filter with Channel and Spatial Reliability [25]. The inherence diagram for CSRT tracker given in Figure 14 below:

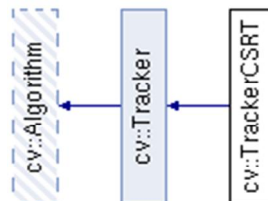


Figure 14. Inference diagram of CSRT tracker in OpenCV library.

OpenCV library is enough big to use different kind of purposes with the support of several programming environment in an Image Processing. This tracker base abstract class for the long-term tracking and it gives state-of-the-art performance with the integration of CNN object detection. Parameters for initialize the tracker with a known bounding box that surrounded the target image returns with true if initialization went successfully, false otherwise. While tracking process reads algorithm parameters from a file storage, where implemented in the Algorithm part of the OpenCV library. Our integrated tracking method updates the tracker with the help of pre-trained object classifier, which includes object features, bounding boxes, object shape information that to find object location in image and class as well and find new most likely bounding box for target from the current frame, which represent the new target location, if true was returned, not modified otherwise. True means that target was located and false means that tracker cannot locate target in current frame. Note, that latter does not imply that tracker has failed, maybe target is indeed missing from the frame, we may say out of sight too.

3.1.5 Experimental Results and Discussion

We have collected our own dataset that contains images taken by drones to make an object detection model to apply for tracking process. Insofar as a main goal was to create a tracker for drone and track only two class of object. For this reason, we collected our own dataset which includes two image classes: pedestrian and vehicle, where we trained our object detection model with the help of Faster RCNN object detection approach to make an object classifier. This step was urgent to create an individual object tracker rather than using open source datasets which has more classes. Furthermore, our trained object classifier dataset images only taken by drone cameras, if we apply our proposed tracking method to drones it works as expected, because of the dataset which contains drone camera-based images that helps while

training object classifier to learn objects location, position, shape, and other parameters of the objects. This difference of dataset's image from ordinary one may influence tracking performance massively. On account of this fact we may see the accomplishment of object classifier even with a small number of images on dataset can show a noticeable outcome while tracking by drone.

After finishing 200K times training our dataset we got our object classifier model and we have tested our object detection (classifier) model work by applying images from different open source resources that accomplishment of our detector demonstrated perfect results, such as shown in Figure 15. We can see the remarkable results of object detection model from different sources; however, the object classifier is not detecting all objects in frame at once in a section © and accuracy rate is under 100%. While tracking we may face thousands of positions, location, shape as well that means it requires more quantity of images on dataset with different position and environment object located images to get perfect results. Nevertheless, with 600 labeled images dataset gave unusual result after training by Faster RCNN object classifier model.

Additionally, object detection model has been tested with different environment, for instance, video sequence that based on drone camera, with real time webcam, and with mobile camera as well, result was good according to the different video environment and situation.

The main goal is in this paper to create a new and simple object tracker for drones and to be simple to set up it to hardware platform. However, we suggested new tracking method by using one of the most popular object detection model integration with a tracker algorithm. Our suggested tracking algorithm to use for a new tracker

is one of the most commonly used algorithms in this sphere and we have integrated detector and tracker.

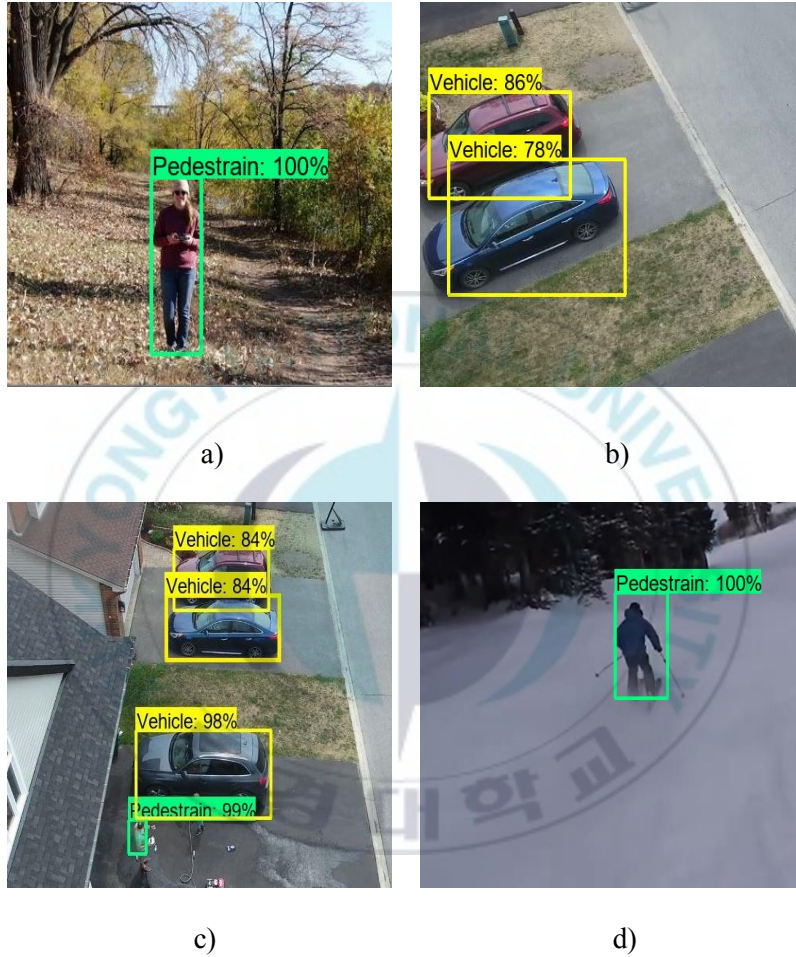


Figure 15. Object detection results of the pre-trained Faster RCNN object classifier:
(a) image from internet sources taken by drone camera; (b, c) picture taken from drone video sequence; (d) image from VOT2018 dataset.

We have tested our tracking method with different open source datasets and got good results and compared with conventional tracker itself. The experiments show

that CSR-DCF tracker algorithm cannot re-establish object once it gone from current frame or tracks object not properly and if the shape or appearance of the tracking object changes in a noticeable drupe tracker can't track object properly and consequently it will lose a target. We have tested this situation on video sequence taken by drone, here is some result of tested video shown in Figure 16 below:

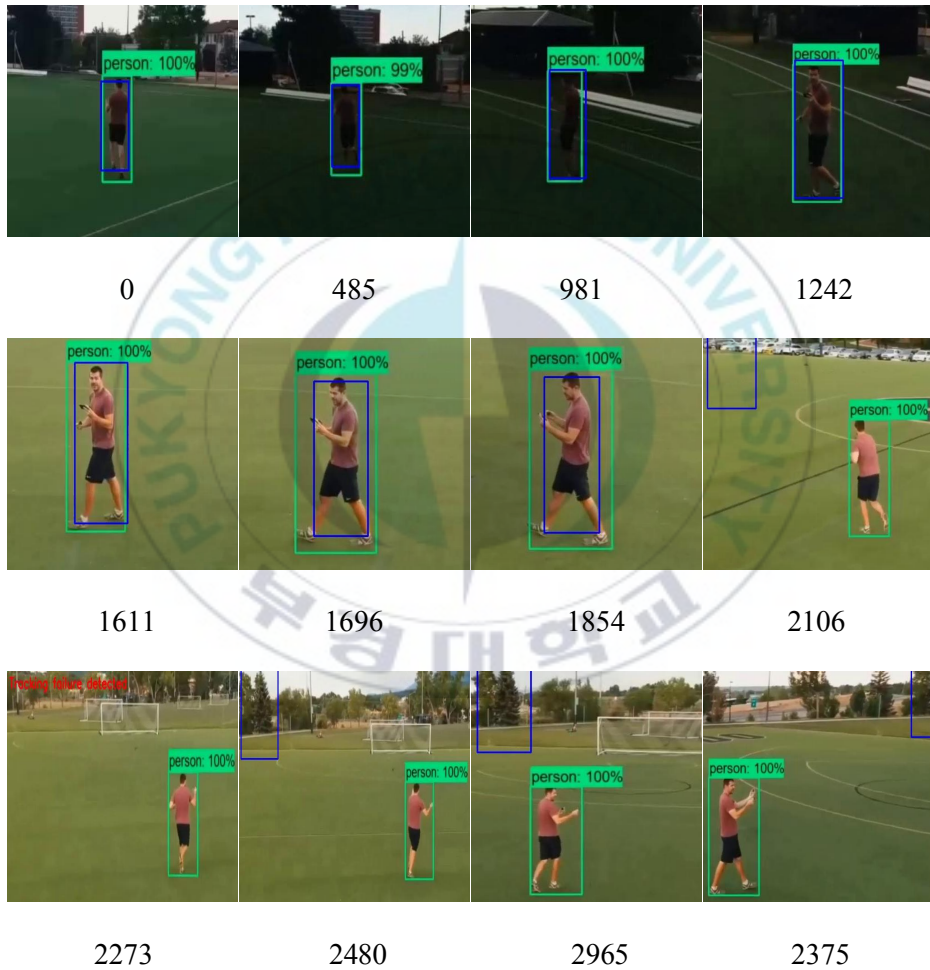


Figure 16. Qualitative results of conventional (CSR-DCF, blue) and proposed (FRCNN_CSRT, green) trackers (0, 485, 981,... - frames).

Initially, conventional tracking method performance was good, it tracked a target properly, but after some movement of target frame by frame and changing a foreground/background color of frame the tracker could not predict properly (given in 1611, 1696, 1854 frames) a new location of target and position of the maximum in correlation between backgrounds and image patch features extraction on position and weight by the channel reliability scores. This video sequence taken by drone after some video footage the target position on video sequence has changed totally where CSR-DCF tracker could not estimate reliability map and channel reliability (frame-2106,2480,2965,3275), even failed at all in some frame (frame-2273). In order to solve that kind of problems we have proposed solution by apply deep learning-based object detector to recovering tracking process in every step of tracking.

Furthermore, a proposed tracking method has been tested with several well-known open source datasets, like OTB (Object Tracking Benchmark), VOT (Visual Object Tracking), and others. However, we have compared results in Table 1 with some open source datasets which conventional method has been tested and posted as a final approach performance in several internet blogs. We can see the comparison result of conventional and proposed tracking methods in Table 1, where has been tested with three different open source datasets that the main difference between two methods is deep learning-based approach and real time frames per second as well.

Table 1. Comparison results of proposed-neural network-based object tracking and conventional methods.

Tracker	Precision-CVPR2013	Precision-OTB100	Precision-OTB50	Deep Learning (Yes/No)	Real Time
CSR-DCF	0.8	0.733		N	Y(13)
Proposed	0.89	0.829	0.81	Y	Y(30)

3.2 The second experiment with proposed method: LSTM network with tracking association

In multi-object tracking, there are many classical problems to overcome across trackable frames. The objective of our work is to create a simple online multi-object tracking platform that would be able to solve some of the basic problems, like appearance similarity, motion, edge detection, and so on. Additionally, we are going to identify classes of trackable objects by taking the last step of the prediction value which follows through with at the end of the sequence. The tracking process is associated with the prediction part of the tracking system that works correspondingly by providing predicted object features, object classes, as well. Our full proposed method is illustrated below step by step in Figure 17:

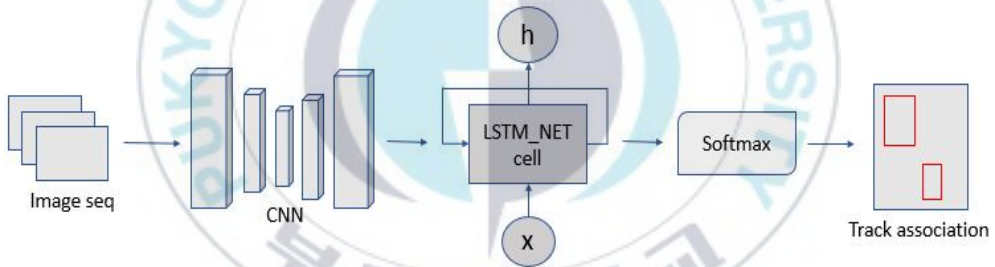


Figure 17. The main steps of the proposed tracking method.

3.2.1 Overview of the proposed multi-object tracking originality

Basically, in Computer Vision there are many image processing studies that related to controlling systems to handle the mess situation of multiple objects movement while visually monitoring. Our proposing method differs from other learning approaches with feature learning strategies of datasets, that takes the labeled images as an input for learning deeply every object inside the frames of the video sequence. At the same time, the original images dataset will be combined to have

clear object features and locations as well as results. As we know labeled images are the process managed by the human-powered task of annotating an image with labels, such as classes, objects locations on an image, and so on. These labels are predetermined by dataset creators (KITTI-tracking and MOT16) are chosen to give it related computer vision model information about what is shown in the image. Our proposal is to applying labeled image datasets into our network by integrating image sequences to get a more accurate outcome model for the feature tracking process. In this proposed network model labeled images are applied as an input, to learn object locations and features by taking center coordinates of the object located bounding boxes configuration and regression, as well as image dimensions configurations into the dense layer of the network that often follows LSTM layers and is used for outputting a prediction, is called Dense, which used instead of fully-connected layer. It's the layer where each neuron of the layer N is connected to all of the neurons $N+1$ from the next layer. It implements the operation $output = X * W + b$ where X is input to the layer, and W and b are actual weights and bias of the layer. A combination of all the bounding boxes center coordinates, dimensions of configuration, and regression in different concatenation layers will be combined to get overall results into one cell memory unit. There are several approaches has been introduced by applying LSTM network for multi-object tracking process, such as MOT with neural gating using bilinear LSTM[27] based on intuitions drawn from recursive least squares, bilinear LSTM stores building blocks of a linear predictor in its memory, which is then coupled with the input in a multiplicative manner, instead of the additive coupling in conventional LSTM approaches. Moreover, a multi-object tracking in videos based on LSTM and Deep Reinforcement learning [28] has been introduced by applying object detection and Markov decision process for sequence decisions to utilizing into deep learning reinforcement. Our model differs from this given recent work with learning strategy, used data structures, tracking proposals as well. Besides, the conventional method of the LSTM network can be applied to different types of

tasks and applications to achieve various results, which is the network cannot be useful conventionally, without the integration of any tracking algorithm.

3.2.2 Proposed LSTM network-based training

Currently, using RNN based methods becoming more common in several research fields of the Computer Vision, such as image/video/audio processing, pattern recognition, biological vision, artificial intelligence, augmented reality, 2D sensors, photography, and so on. The extensive use of RNN is due to their working ability of memory unit's direct control states, such as gate state, gate memory and forget gates that work in a certain time and without interruptions. In this section, we are going to describe our LSTM based network model for tracking purposes. In this work, the main idea is to create a network that to learn information directly from featured datasets that contain labeled images, ground truth info, sequence info, detection coordination of the trackable objects details for training procedure. The approach of training operation goes offline for learning features, estimating objects movement orientation to get proper output for integrating with tracking association. LSTM network has its capability to exploit object characteristics to overcome classification problems, and other typical challenges, such as occlusion, object similarity, multiple interconnected objects, and so on. Proposing architecture of the LSTM network builds on multiple network layers to utilize the learning model to be more productive to exploit features of the dataset images for getting better performance with the associated tracking method. To testify our approach, we took datasets that commonly used with different methodologies. They are KITTI tracking [29] and MOT16 [30], these datasets are different taken inside and outside environment content videos that include single and multiple object scenes of human and vehicles.

3.2.2.1 Proposed network model

The proposing LSTM based network model shown in Figure 18 below that has been trained with two various types of datasets individually, to use trained output results with tracking association parallelly. Our offline trained output results give us the initial estimation of prediction and to identify the object class for tracking procedure. The training process starts by configuring dataset details with training options that could integrate the whole process correctly. KITTI-tracking training data set contains 21 video sequences that labeled into two class images. All images have different scenery of vision and viewpoints.

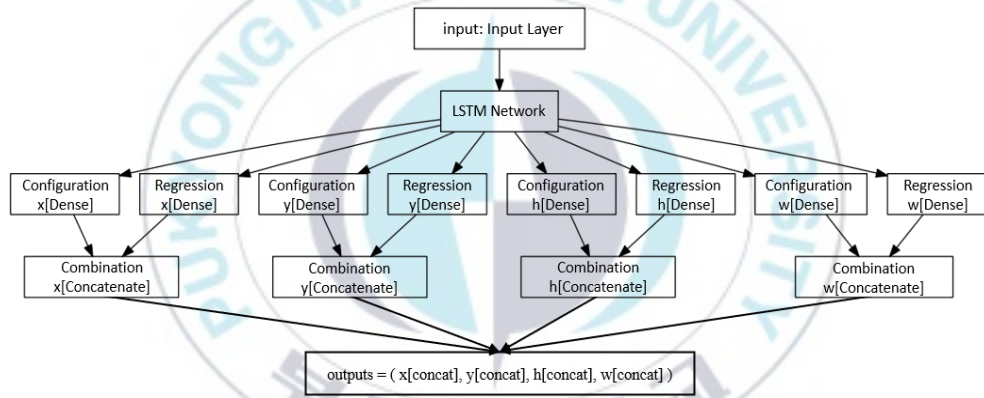


Figure 18. Proposing LSTM Network model structure.

The LSTM network model takes the values of the sequence images to learn directly from labeled images for deeply excavate the features of the images by obtaining a certain period of the sequence for deeper exploration of the appearance similarity from the sequential images. At the same time, the network capable of estimating the position of the trackable objects with the help of the input parameters of the dataset, such as labeled images, a region of the interest (position of the multiple objects in the frame), and so on. Figure 18 above illustrates the purposed LSTM network structure that shows the detailed steps of the learning model. Here we can

see the network is taking the configuration of the object located coordinates and values of the regression to learn the features, predicting trackable object classes as well. Moreover, the network can save a predicted object location by the rules of the LSTM cell state units. The necessary steps of the predicting unit and memorizing follows in every frame of the video sequence by learning features parallelly. After receiving a trained model output in offline mode, we are going to use it as an initial feeding example to start the tracking process, also will be used for evaluation of our model.

3.2.2.2 Building LSTM network

To build our network we started by integrating the feature extraction part of the network with the training LSTM Network model. Therefore, we have created a simple feature extractor to provide the network with bounding box information to learn the dataset properly, provided with an object position map that contains the location of the target, a region of interest, and so on. The extracting features of the images and bounding boxes help to learn the searching area of the interests and make it smaller, besides, it maintains the network from overbalancing feature details. The network created on the Keras model that includes libraries and all useful modules, such as layers, configurations, and so on. The prediction section of the training model gets information from trained model features, bounding box learning section, and ext. The loss function of training mode indicates the result of a bad prediction where a number indicating how bad the model's prediction was on a single example. In our model we used the Huber loss function that changes the distance point from a quadratic to linear, also applied SoftMax cross-entropy to identify several label classes to calculate losses. Optimization problems considered as an analyzing action to solve a problem for some sort of data and reduce the set of model selection and validation while learning process. In this work, the adaptive moment estimation (Adam) optimizer used to minimize the cost function in training a neural network.

Other metrics such as prediction, labels, and accuracy measures have been calculated by taking input details from learned features and labels information.

3.2.2.3 Training configuration

The evaluation of the training process relies on the calculation of prediction metrics, losses, labels, the accuracy of prediction and labels, precision, and ext. The training process goes offline mode, where all required options and input datasets set to a necessary part of the assistant. A network model trained the model 100 epoch times to get trained model output, graph output, and evaluation files as well. The model has been tested with two common datasets individually, where a training process holds on individually and received outputs in a different location. The learning rate has been set as 0.001 manually in case of exploring the object features deeply, when the learning rate is small the training will take a longer time than as usual. The training and estimation process will be proceed parallelly into one action while training and shows the results of evaluation every 1000 iteration, which includes training evaluation precision, accuracy, loss, and other outcomes. Besides, the results of training outcome files will be stored in the provided location. The estimated results of the training process show the model productivity, learning ability, and mutual compatibility of the model.

3.2.3 Proposed LSTM tracking association

The multi-object tracking process is a more complicated procedure comparing to observing a single object in the video frame. The LSTM network model has been associated with the tracking process by integrating tracking facilities to the network with the help of the prediction ability of LSTM cell state units. The object tracking process provided with extracted features that contain an object located bounding boxes to track with a high accuracy rate and clearly. However, we have tackled some misleading and uncertain multiple object appearance while using detection-based tracking, but our proposed tracking association method gave us to track every object

as an identical object with unique IDs. Additionally, the identification process holds separately one by one and solving other detection-based problems parallelly. In case of tracking a single target track the state space that location of the bounding box and their height and width has been applied. As a starting point for tracking, we took a shape of the objects and classes by getting the covariance matrix of the initial state, and this distribution was an essential step for giving them a unique track identity. Additionally, identifying every object as one identical among multiple objects in a frame, while to learn and remember several time steps tracks by the network took more time. Multi-target tracking differs visibly from a single target tracking process, which we should take into consideration that to apply our trained LSTM model while integrating with a massive object tracking process. We have an LSTM graph file that created while training our model, which used for predicting the next target by running the graph runner unit. There is a list of active tracks at the current time step with the number of object classes in a one-time step sequence to be tracked. To obtain the next prediction from each track, it returns an array of the shape with some tracks and classes as well. The following Figure 19 illustrates the structure of the tracking associated with the LSTM model:

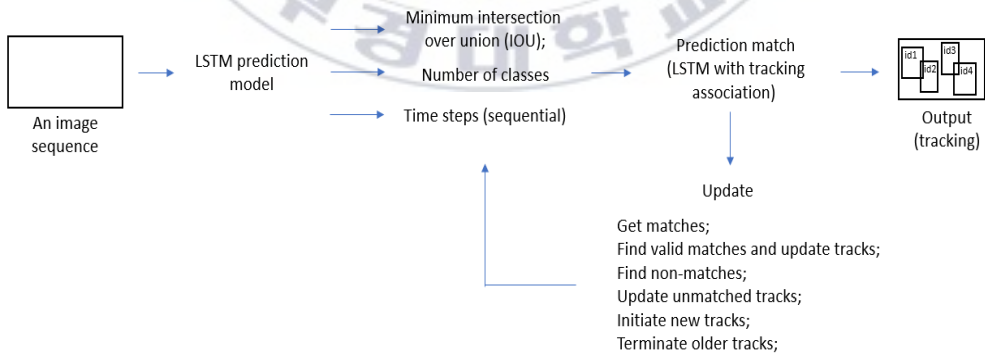


Figure 19. Tracking association integration with the LSTM model.

3.2.4 Experiment results and discussion

For exploring the proposed network capability, adaptability, and working stability with a different situation, the network tested with two different opensource datasets. The training process is conducted in offline mode, by applying KITTI-tracking and MOT16's training set around 13K images which means: in the KITTI-tracking training set has 20 video sequences, around 8K images; in the MOT16 training set 7 video sequences around 5.3K images has trained. These datasets have in different position multi and single target located video sequences that help to test the network learning and prediction facilities by training and testing it with a test set of the datasets. Our experimental results presented better results compared to other MOT researches, yet, we have some minor drawbacks like time consuming and unstable accuracy issues while training, training losses, and not much high accuracy results from other methods. This evaluation below can show only the results of the multi-object tracking performance of our proposals and other techniques. As we mentioned in previous sections, there is not much LSTM based multi-object tracking papers available, even in existing ones also used different type of opensource datasets, structures, and for other purposes. Therefore, we have faced some shortcoming comparison difficulties for evaluating our model with other conventional methods. Although, we tried to be accurate and discrete in every comparison measurement evaluation.

3.2.4.1 Training evaluation.

Evaluation of the training is like a systematic process that to analyze a training program and initiatives whether if it's effective and efficient, and the possibility of working with different class datasets. We have evaluated our training process by calculating the network capability, which includes training accuracy, loss, precision accuracy, mean of interconnected over union (Mean IoU), and accuracy of evaluation. A given Table 2 below shows the training evaluation results:

Table 2. Training evaluation of LSTM network model.

Number of Iteration	Train Accuracy (%)	Loss (%)	Precision Evaluation (%)	Mean IoU Evaluation (%)	Accuracy Evaluation (%)
1(1257)	41.1	23.8	51.2	60.4	38.9
2(29318)	53.4	10.3	67.7	67.6	59.12
3(78200)	71.1	2.1	85.3	71.6	88.29
4(133478)	67.8	1.09	76.8	70.3	60.18
5(198321)	82.6	0.05	90.4	82.9	78.37
6(242412)	76.3	1	96.7	86.1	70.23

Given results on this table above shows a randomly taken iteration steps outcome while training the network. The last column is the final iteration number with training results. In this table, the train accuracy column is the calculation of training quality which means how our proposed LSTM network performed well and did not overfit. The loss column of the table shows that how much percentage data has not been learned or skipped by our network model while training process. A precision evaluation unit of the table indicates the accuracy of the LSTM network prediction. Mean Interconnected over Union section part shows how the network performed while learning the interconnected objects in every image. The last column of the training evaluation of the table describes the overall performance of the training process.

3.2.4.2 Tracking evaluation.

The following unit of the experiment results represents a tracking association performance with the LSTM network model. In the case of creating a multi-object tracking model, we have integrated a tracking association with the LSTM network model. The entire system can work in an online mode. LSTM network-based multi-object tracking model has been tested with different types of open-source datasets, to evaluate the model performance in a different situation. Following Table 3 shows multi-object tracking evaluation on KITTI-tracking dataset:

Table 3. LSTM network based multi object tracking evaluation on KITTI tracking dataset.

Video Sequence ID	Number of frames	MOTA (%)↑	MOTP (%)↑	MT↑	ML↓	IDs	Number of matched↑	Number of objects
0000	153	76.4	79.1	15	20	2	623	708
0004	313	81.2	84.03	41	98	192	900	1108
0008	389	84.8	86.77	28	119	192	1157	1365
0013	339	91.8	92	68	23	101	1353	1474
0016	208	97.4	96	28	12	72	3042	3122
0019	1058	96.8	97.12	106	85	251	8538	8820

Table 3 above shows the results of the LSTM network-based multi-object tracking on the KITTI tracking dataset. We took the video sequence IDs randomly from the KITTI tracking dataset, that every video sequence has a different number of frames as well as in a different scenario. Moreover, in this table given a result of multi-object tracking accuracy (MOTA), multi-object tracking precision (MOTP), tracked (MT) objects ratio, and mostly loss (ML), and so on. The number of matched columns describes the same targets has been tracked on the frame, which means one single target or similar target has been tracked several times.

3.2.4.3 Comparison results.

To explore the performance of the proposed tracking method we have compared the tracking results with other method's outcomes. The comparison results are given below in tables 4 and 5 which describe two different opensource dataset testing results with different MOT methodologies. Lack of resources in LSTM based MOT study, we have added some other methodology results. Nonetheless, tried to put RNN based works to compare and evaluate our model properly in a multi-object tracking procedure. The given Table 4 below indicates a comparison results of multi-object tracking on KITTI tracking dataset:

Table 4. Comparison of our method performance on KITTI-tracking dataset with recent works.

Methods	Tracking mode	MOTA↑(%)	MOTP↑(%)	MT↑(%)	ML↓(%)	IDs
MDP	online	76.59	82.10	52.15	13.38	130
LP-SSVM	online	77.63	77.80	62.61	8.76	62
NOMT	online	78.15	79.46	57.23	13.23	31
MCMOT-CPD	online	78.90	82.13	52.31	11.69	228
JCSTD	online	80.57	81.81	56.77	7.38	61
LSTMNET-TA (ours)	online	88.06	89.22	59.12	10.42	135

A given Table 4 above indicates a comparison result of tracking methods on the KITTI tracking dataset, within new column feature - tracking mode, from this given result we can say that our proposed LSTM network-based multi-object tracking model has achieved one of the best results among given recent multi-object tracking methods, in most basic performance quality features (MOTA, MOTP, MT).

The following itinerary information below describes a comparison results of multi-object tracking methods on the MOT16 dataset:

Table 5. Comparison of our method performance on MOT16 dataset with recent works.

Methods	Tracking mode	MOTA↑(%)	MOTP↑(%)	MT↑ (%)	ML↓(%)	IDs
RNN LSTM [31]	online	19.0	71.0	5.5	45.6	1490
MDP	online	30.3	71.3	13.0	38.4	680
JPDA	online	23.8	68.2	5.0	58.1	365
Response	online	62.0	73.6	37.7	20.7	909
DPT_DPT	online	61.3	79.1	32.1	18.6	739
LSTMNET-TA (ours)	online	69.12	75.17	45.26	17.18	810

Table 5 shows the results of multi-object tracking comparing with the most recent works on the MOT16 dataset. The numerical results on this table show the average percentage of performance on the MOT16 dataset with different multi-object tracking approaches. The percentage of accuracy rate on this table in the first place with 69.12, but in the precision column, our method shows second results. Although, LSTMNET_TA model performance of MOT accuracy and precision was higher than RNN_LSTM based MOT. Moreover, our model presented good results in different measurements.

The main goal of this work is to implement a new multi-object tracking technique that should be capable to track multiple targets at the same time with IDs. Our multi-target tracking approach showed that the LSTM network-based tracking association can work and adapt easily with new targets. Visual qualitative results of the KITTI-tracking dataset tested results of the proposed multi-object tracking method given below in Figure 20:

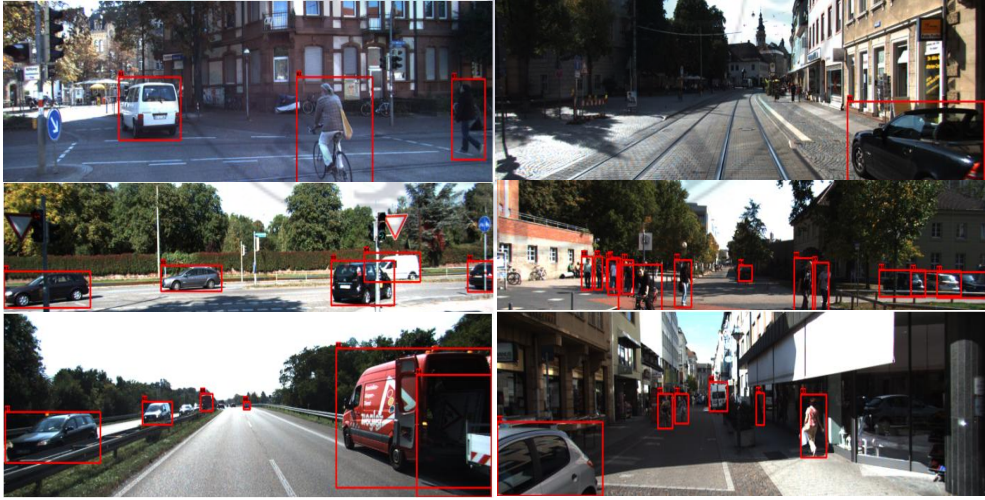


Figure 20. Qualitative results of the proposed method based on KITTI tracking dataset.

IV. Conclusion

In general, this thesis is divided into two parts. In the first part of thesis object tracking by detection method has discussed. The method proposed in this part proves that it can reach perfect performance in tracking process using deep learning-based object detection approach that could achieve better results in several tested open source datasets. We have presented Faster RCNN training procedure only with two class object classification includes 600 images. Our proposed additional object classifier part to CSR-DCF algorithm could help to overcome problems that mentioned in abstract and through this manual script. In this method we have integrated deep learning-based object classifier model with CSRT tracker OpenCV implementation version of CSR-DCF algorithm with the support of OpenCV DNN module. Comparison results showed a much better performance in proposed tracking method that in tracking process an object classifier gives exact location of the target in frame with the benefit of pre-trained object classifier. Moreover, because of the drone dataset which all training and testing images taken from drone camera could hand even with not much pre-trained images. The method presented here can be used by modifying for any kind of intended controlling systems or tracking purposes that can be managed easily.

In the second part of the experiment, we have implemented LSTM network-based model for tracking process, instead of using object detection part for object tracking. The strategy of learning and predicting objects without detection part showed that LSTM network-based target tracking has more capability and adaptation rather than other approaches. This method performance is reliable and has more advantages and distinguishes from other types of neural networks. Such as memory cells, controlling gates, working ability within time series video sequences, adopting easily with any kind of situation, and so on. Besides, the network has prediction skills and memorizing it into cells in a time series period. In this study, we have presented

our network model that extracts features and learns as an object. As we mentioned in the previous stages of our description above, LSMT is one of the RNN based structures that can be directed to any kind of purpose. So, there are not many works available on the MOT field with the LSTM model, also, there is not much outcome to compare with. We tried to make a specific comparison with RNN_LSTM [31] based MOT, which our model showed better results. The entire process can work in online mode, integration of tracking association supports tracking multi targets easily by taking detected and predicted bounding boxes from the LSTM network unit. Comparison studies demonstrated that our presented network model along with tracking association multi-object tracker performance is much better than other neural network-based techniques. Moreover, our system is not complicated to set in a real project application. This presented multi-target tracking system can be applied any kind of required filed of tracking purposes, because of its adaptability and also can be managed easily.

In the future work, we are going to improve the network configuration and compatibility with other additional properties, additionally, accuracy and performance of both methods for applying different situations to achieve more computable outcome in real time applications.

References

- [1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006.
- [2] A. Ali, A. Jalil, J. W. Niu, X. K. Zhao, S. Rathore, J. Ahmed, and M. Aksam Iftikhar, "Visual object tracking-classical and contemporary approaches," *Frontiers of Computer Science*, vol. 10, no. 1, pp. 167-188, Feb. 2016.
- [3] A.-H. A. El-Shafie and S. E. D. Habib, "A Survey on Hardware Implementations of Visual Object Trackers", *arXiv:1711.02441* 2017.
- [4] P. Viola and M. Jones, "Robust Real-time Object Detection," *Second international workshop on statistical and computational theories of vision – modeling, learning, computing, and sampling*, Vancouver, Canada, July 13, 2001.
- [5] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *INRIA Rhone-Alps*, 655 avenue de l'Europe, Montbonnot 38334, France, 2005.
- [6] A. Krizhevsky, I. Sutskever, Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks".
- [7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," 22 Oct. 2014.
- [8] R. Girshick, "Fast R-CNN," 27 Sept. 2015.
- [9] R. Girshick, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," 6 Jan. 2016.
- [10] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional Siamese networks for object tracking," in *European Conference on Computer Vision (ECCV)*, 2016.

- [11] A. Milan, S.H. Rezatofighi, A. Dick, I. Reid, and K. Schindler “Online Multi-Target Tracking Using Recurrent Neural Networks,” *Association for the Advancement of Artificial Intelligence*, pp.4225-4232, Feb. 2017.
- [12] Sh. Sun, N. Akhtar, H. Sh. Song, A. Mian, and M. Shah, “Deep Affinity Network for Multiple Object Tracking,” *Journal of latex class files*, Vol. 13, No. 9, Sept. 2017.
- [13] Object detection https://en.m.wikipedia.org/wiki/Object_detection.
- [14] S. Phon-Amnuaisuk, Ken T. Murata, P. Pavarangkoon, K. Yamamoto, and T. Mizuhara, “Exploring the Applications of Faster R-CNN and Single-Shot Multi-box Detection in a Smart Nursery Domain,” *arXiv:1808.08675v1* [cs.CV] 27 Aug. 2018.
- [15] Gonzalez, R.C., and Woods, R.E.: *Digital Image Processing*. Pearson (4 edition) 2017.
- [16] Faster RCNN, <https://towardsdatascience.com/faster-rcnn-object-detection-f865e5ed7fc4>
- [17] Region Proposal Network (RPN) – Backbone of Faster RCNN, <https://medium.com/egen/region-proposal-network-rpn-backbone-of-faster-rcnn-4a744a38d7f9>
- [18] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 6, pp. 1137–1149, June 2017.
- [19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, Las Vegas, NV, USA, June 2016.
- [20] A. A. Butt and R. T. Collins, “Multi-target tracking by Lagrangian relaxation to min-cost network flow,” in *Proceedings of the IEEE Conference on*

- Computer Vision and Pattern Recognition (CVPR)*, pp. 1846–1853, Portland, OR, USA, June 2013.
- [21] S. H. Bae and K. J. Yoon, “Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 40, No. 3, pp. 595–610, Mar. 2018.
 - [22] Wikipedia, https://en.wikipedia.org/wiki/Long_short-term_memory
 - [23] Benjamin J. Radford, Leonardo M., Apolonio, Antonio J. Trias, and Jim A. Simpson, “Network Traffic Anomaly Detection Using Recurrent Neural Networks”, *arXiv:1803.10769v1 [cs.CY]* 28 Mar. 2018.
 - [24] Understanding the impact of learning rate on neural network performance, <https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/>
 - [25] A. Lukezic, T. Vojir, L. C. Zajc, J. Matas, and M. Kristan, “Discriminative Correlation Filter Tracker with Channel and Spatial Reliability,” *International Journal of Computer Vision: Volume 126, Issue 7*, pp 671–688, July 2018.
 - [26] OpenCV dnn module, `cv::dnn::Net` Class Reference, https://docs.opencv.org/master/db/d30/classcv_1_1dnn_1_1Net.html
 - [27] Kim Ch., Li F., James M. Rehg “Multi-object Tracking with Neural Gating Using Bilinear LSTM” European Conference on Computer Vision (ECCV), 2018, pp. 200-215.
 - [28] Ming-xin J., Chao D., Zhi-geng P., Lan-fang W., Xing S. “Multi-Object Tracking in Videos Based on LSTM and Deep Reinforcement Learning”.
 - [29] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The KITTI vision benchmark suite,” Conference on Computer Vision and Pattern recognition (CVPR), 2012.

- [30] A. Milan, L. Leal-Taix'e, I. D. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," CoRR, vol.abs/1603.00831, 2016.
- [31] Anton M., S. Hamid R., Anthony D., Ian Reid, Konrad S. "Online Multi-Target Tracking Using Recurrent Neural Networks" Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), 2017.

