CarNet: MULTI VEHICLE TRACKING FROM AERIAL UAV IMAGES

A PREPRINT

Arman Nikkhah [*]
Department of Computer Science
Github :github.com/IamArmanNikkhah

Mahdi Asefi
Department of Computer Science
email:mahdiasefi1379@gmail.com

ABSTRACT

Keywords First keyword · Second keyword · More

1 Introduction

2 Object Detection

Object detection is one of the most important aspects of this project because the quality of detections has a significant impact on tracking performance .Bewley et al. [2017]. We know that Using only detection algorithms is not a good solution for this problem because there are different challenges for tracking performance. the clearest challenge of this technique is an occultation. The occlusion of objects in videos is one of the most common obstacles to the seamless tracking of objects. For example, an object's path may intersect the tree and the detector fails to locate the object. See Section 2.

2.1 YOLO: you only look once

YOLO, which has been proposed by Redmon et al. [2016], is an extremely fast and accurate object detection architecture and subsequently improved in 2016 (YOLOv2 Redmon [2017a]) and in 2018 (YOLOv3 Joseph Redmon [2018]).It is so fast that it can run in real time on a video. YOLO uses Fully Convolutional Networks that was first introduced in a 2015 paper by Jonathan Long [2015], for semantic segmentation. The authors pointed out that you could replace the dense layers at the top of a CNN by convolutional layers. The advantages of this approach is that while a dense layer expects a specific input size (since it has one weight per input feature), a convolutional layer will happily process images of any size so it can process any given input size. Convolutional networks with three or more layers were actually inspired by the architecture of the visual cortex [LeCun et al. 1990]. The first two layers are a multi-scale filter bank that maps the input to a space where neurons can be sensitive to various kinds of features such as edges or corners.

2.2 YOLO v1

YOLO divides the input image into S×S grids.Then it produces a bounding box for each grid cell. for example, a (5×5) grid will generate 25 bounding boxes. Each bounding box is represented as a 5-tuple (class, minx, miny, maxx, maxy) containing the coordinates of its minimum and maximum values in the x and y dimensions.Bounding boxes are used to create a rectangle so that the object detector can detect foreground objects such as cars. If the center coordinate of the GT (Ground Truth) of an object falls into a grid, the grid is responsible for detecting the object. The innovation of YOLO is that it reforms the Region Proposal detection framework: RCNN series need to generate Region Proposal in which to complete classification and regression. But there is overlap between Region Proposal, which will bring a lot of repetition work. However, YOLO predicts the bounding box of the object contained in all grids, the location reliability, as well as the probability vectors of all classes at one time, thus it solves problem one-shot.[Lu et al., 2018]

[*]Computer Science Student (*email:* arman.nikkhah.79@gmail.com)

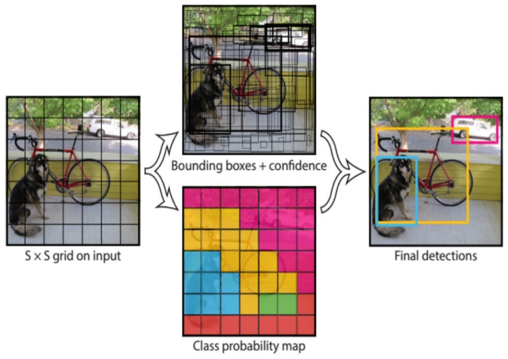


Figure 1: YOLO model detection as a regression problem. it devides the image into S×S grid and for each grid cell predicts *B* bounding boxes, confidence for those boxes and *C* class probabilities.

YOLO network uses 1x1 convolutional layer (for cross-channel integration of information) and 3x3 convolutional layer compared to simply using the Inception module. YOLO v1 network structure consists of 24 convolution layers and 2 full connection layers that are responsible for the classification of objects and regression of bounding boxes. The final output is a 7 x 7 x 30 tensor, as shown in Figure 2. Leaky ReLU activation is used for all layers except the final layer. The final layer uses a linear activation function.Leaky ReLU activation has been found to be a very effective activation function, especially for deep neural networks.[Xu et al., 2015]

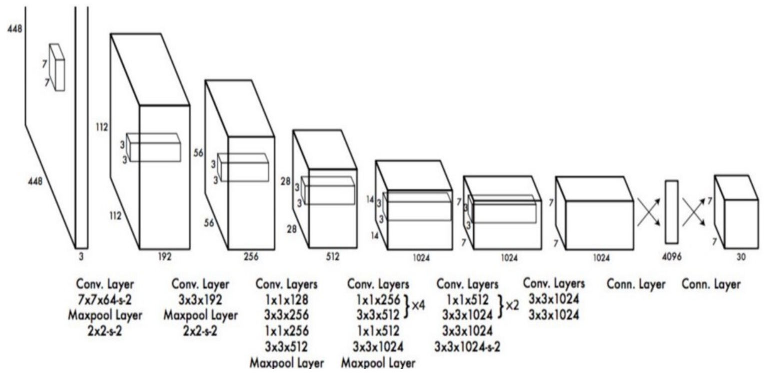


Figure 2: Yolo Architecture

YOLO v1 has some important drawbacks that can cause serious trouble in our project. One of these weaknesses is Bad performance when there are groups of small objects because each grid can only detect one object. In our project, because the goal is to detect and track lots of small vehicles and the distance between them is also so small we cannot use YOLO v1 as the main frame for detection. see Figure 3

2.3 YOLO v2

Comparing YOLO to Fast R-CNN, we found that YOLO exhibits a substantial number of localization errors.(see Figure 4) In addition, YOLO has relatively low recall compared to region proposal-based methods. Thus we focus mainly on improving recall and localization while maintaining classification accuracy[Redmon, 2017b].Therefore, the main improvements of YOLO v2 include:

- BN (Batch Normalization): Batch normalization allows for significant convergence improvements without requiring other regularization methods.(Redmon [2017a]) By adding BN layer after each layer, the entire batch data can be normalized to a space with a mean of 0 and variance of 1, which can prevent the gradient from disappearing as well as gradient explosion, and make network convergence faster.[Ioffe and Szegedy, 2015]
- Anchor boxes: In version 1 of the YOLO algorithm, the full connection layer is used to predict the coordinates of bbox directly after the convolutional layer. YOLO V2 eliminates the connection layer using Faster R-CNN

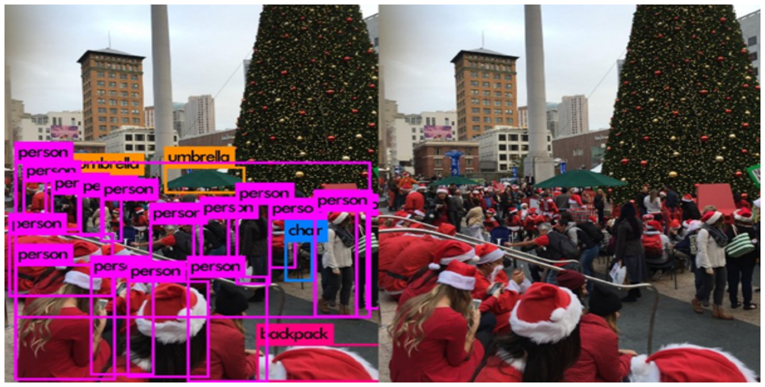


Figure 3: From the above image, we can see that YOLO version1 have limitation based upon the closeness of object. As we can see, YOLO is detecting only 5 Santa's from lower left corner, but there are 9 Santa's.

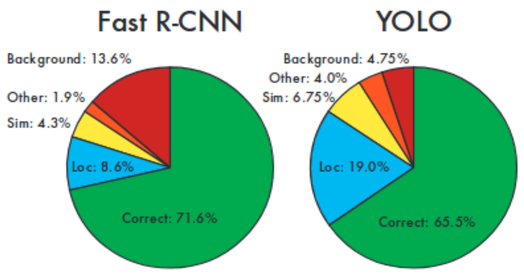


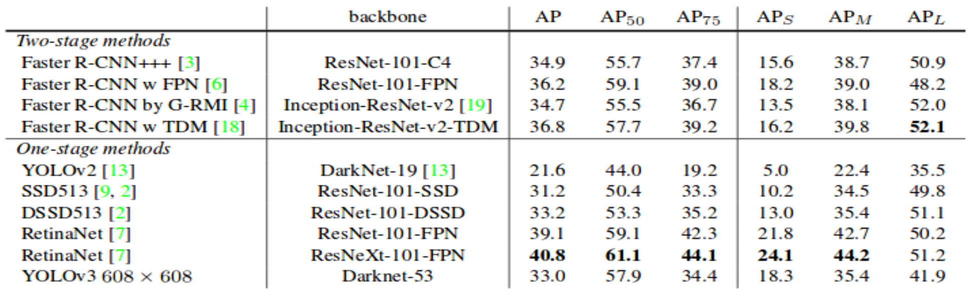
Figure 4: comparing YOLO v1 and R-CNN accuracy[Redmon et al., 2016]

idea and introduces Anchor Boxes, which improves recall rate significantly.Recall rate is a measure of how accurate a network is for a given set of input data.

- Multi-scale training: While YOLO v1 maintains a fixed input image size for its training network, YOLO v2 randomly changes the image size every 10 epochs .This regime forces the network to learn to predict well across a variety of input dimensions. This means the same network can predict detections at different resolutions. In short, YOLOv2 offers a good balance between speed and accuracy due to its ability to run faster at smaller input sizes.[Redmon, 2017b]

2.4 YOLO v3

There was an incremental improvement over YOLOv2 network to improve performance of localization. YOLOv3 gives a MAP of 57.9 on COCO dataset for IOU 0.5. For a comparison, see the Figure 5:



	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [3]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [6]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [4]	Inception-ResNet-v2 [19]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [18]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [13]	DarkNet-19 [13]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [9, 2]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [2]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [7]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [7]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608×608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Figure 5: Comparison of mean precisions, 2018

Now we want to look what improvements have been made in this version:[Joseph Redmon, 2018]

- Class Predictions : Instead of softmax layers, Yolov3 uses independent logistic classifiers for each class. This process is used to create a multi-label classification.
- Predictions across scales : YOLOv3 predicts boxes at three different scales to support detection at different scales. Using a method similar to feature pyramid networks, features are then extracted from each scale.
- Feature Extractor : While YOLOv2 used Darknet-19 as its backbone, YOLOv3 uses a new network called Darknet-53. Darknet-53 has 53 convolutional layers, it is deeper than YOLOv2 and it also has residuals or shortcut connections. Compared to Darknet-19, and ResNet-101 and ResNet-152 it is significantly more powerful and efficient.

These improvements caused Precision for small objects to improve and it is now better than Faster RCNN, but Retinanet is still better in this respect. In addition to this as a result of adding a feature pyramid-like method, predictions at different scales or aspect ratios for the same object improved.

2.5 YOLO v4

YOLO v4 incorporates several BoS (bag of specials) and state-of-the-art BoF (bag of freebies). While the BoF improves the detector's accuracy, it does not increase the inference time. The cost of training is only increased. Alternatively, the BoS increase inference costs slightly, but improve accuracy of the object detection.[Bochkovskiy et al., 2020] Performance and speed improved slightly as a result of these changes.see Figure 6

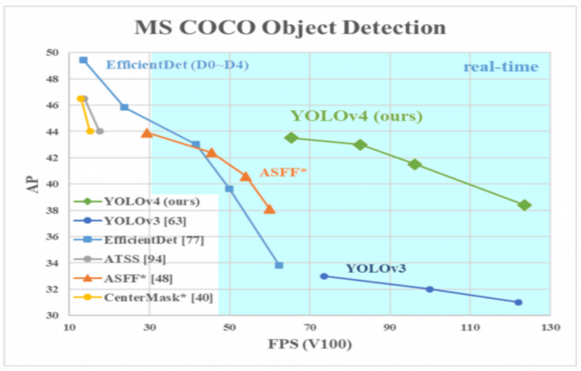


Figure 6: YOLO v4's speed and accuracy

2.6 YOLO v5

A major difference from all previous releases is that Yolo v5 uses PyTorch instead of forking the original Darknet code. Similar to YOLO v4, YOLO v5 has a CSP backbone and a PA-NET neck. The main enhancements are the augmentation of mosaic data and the auto-learning anchors for bounding boxes.In this project we use *Ultralytics* framework to train our object detector.see Figure7

Figure 7: *Ultralytics* framework

3 Data

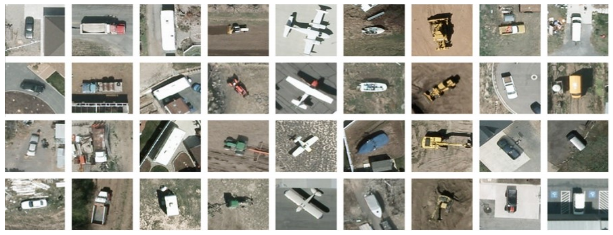
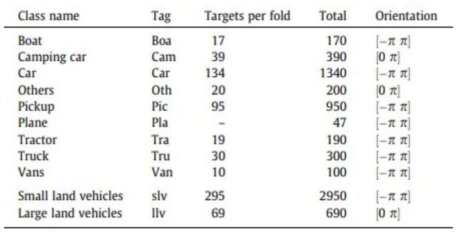
On aerial images, the performance of the classifier trained on a traditional dataset is poor since the aerial images include the following unique features[Lu et al., 2018]:

1. **High level of background complexity**: In aerial images, there is a large field of view (typically a few square kilometers), and backgrounds may be a common occurrence, which will interfere with the detection of objects.Some of the things in the image, for example, may resemble the vehicles we're looking for.
2. **Small objects**: Aerial pictures' objects are usually only a few hundred or even a few pixels in size, therefore the quantity of information they provide is likewise limited. Vehicles in VEDAI are 20x40 pixels in size.
3. **Scale diversity**: UAVs may shoot from tens of meters to kilometers in altitude, resulting in a wide range of object sizes on the ground.

It is generally difficult, for the reasons above, to train an optimal classifier for aerial picture object recognition tasks using traditional datasets.As a result, a unique aerial image dataset is required. Two publicly available aerial image datasets are combined and processed in this study to create a new aerial image dataset suited for training.

3.1 VEDAI

Sebastien Razakarivony and Frederic Jurie of University of Caen have made the VEDAI (Vehicle Detection in Aerial Imagery) dataset. The original data comes from Utah AGRC's public database, which gives access to many freely distributable aerial orthonormal images.A total of 1,210, 512 × 512 images have been manually selected. All of the images are taken with a same height and a same angle, the dataset contains nine categories of objects: airplane, boat, car, truck, tractor, camping, pickup, vans, and others, the number of every class as shown in Figure 8. The VEDAI dataset features an average of 5.5 vehicle objects, which account for only 0.7% of the total pixels in images.[Razakarivony and Jurie, 2016]

Class name	Tag	Targets per fold	Total	Orientation
Boat	Boa	17	170	$[-\pi\ \pi]$
Camping car	Cam	39	390	$[0\ \pi]$
Car	Car	134	1340	$[-\pi\ \pi]$
Others	Oth	20	200	$[0\ \pi]$
Pickup	Pic	95	950	$[-\pi\ \pi]$
Plane	Pla	–	47	$[-\pi\ \pi]$
Tractor	Tra	19	190	$[-\pi\ \pi]$
Truck	Tru	30	300	$[-\pi\ \pi]$
Vans	Van	10	100	$[-\pi\ \pi]$
Small land vehicles	slv	295	2950	$[-\pi\ \pi]$
Large land vehicles	llv	69	690	$[0\ \pi]$

Figure 8: VEDAI dataset[Razakarivony and Jurie, 2016]

3.2 VAID

VAID data set contains 5985 1137 x 640 images captured under different traffic conditions, and annotated with 7 categories of objects : sedan, minibus, truck, pickup truck, bus, cement truck, trailer(Figure 9).more information about objects is shown in Table 1 .[LIN et al., 2020].

Location	Image	Sedan	Minibus	Truck	Pickup truck	Bus	Cement truck	Trailer
University Campus	3,257	11,385	406	605	611	292	17	36
Urban Area	1,118	18,349	95	1,014	822	225	6	10
Suburb	1,610	10,596	0	1,568	1,578	63	168	758
Total	5,985	40,330	501	3,187	3,011	580	191	804

Table 1: Some statistics of VAID dataset.

A drone was used to take the photos in the dataset. During video recording, the drone's altitude is maintained between 90 and 95 meters above ground to maintain consistency across all images. The output resolution is 2720 × 1530 at 2.7K and the frame rate is about 23.98 fps. The apparent size in the image for an ordinary car with a length of 5 meters and a width of 2.6 meters is around 110*45 pixels. Using the VAID dataset, the images are scaled to a resolution of 1137*640, and the sedans are approximately 40 x 20 pixels.[LIN et al., 2020]

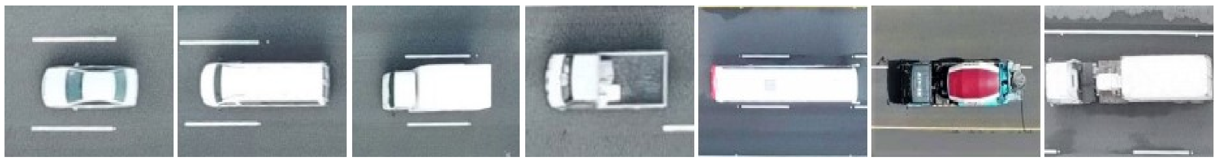


Figure 9: In VAID dataset, the common vehicles are classified to 7 categories, namely (a) sedan, (b) minibus, (c) truck, (d) pickup truck, (e) bus, (f) cement truck and (g) trailer. The sample images are shown in the figure from the left to the right accordingly

3.3 DOTA

Aerial image dataset DOTA (Dataset for Object detection in Aerial images) was created by Professor Xia Guisong of Wuhan University and Professor Bai Xiang of Huazhong University of Science and Technology.[Xia et al., 2019] DOTA comprises a total of 2806 pictures with a resolution of 4000 x 4000 pixels that have been carefully tagged into 15 example classes.These classes of samples are as follows : plane, ship, storage tank, baseball diamond, tennis court, basketball court, ground track field, harbor, bridge, large vehicle, small vehicle, helicopter, roundabout, soccer ball field and swimming pool.see Figure 10

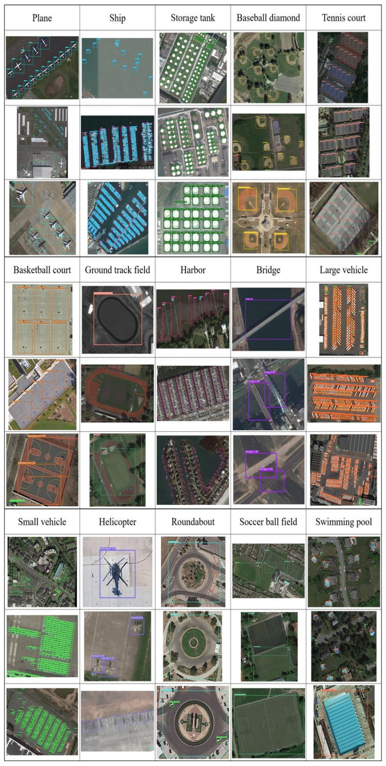


Figure 10: DOTA dataset

3.4 Data Preparation

For Training the YOLO model, the standard dataset consists primarily of two parts: images and labels. Images are JPEG files and labels are TXT files.The labels record annotations of the samples in the corresponding images. Annotations are formatted as follows:

$$(class, x, y, width, height)$$

Where (x, y, w, h) are normalized values.An overview of the three public aerial image datasets is given in Table 2 .

Dataset	Image Format	Images	Classes	Image size
VEDAI	JPEG	1,250	9	512×512
VAID	JPEG	5,985	7	1137×640
DOTA	JPEG	2,806	15	about 4000×4000

Table 2: The basic information of the three public aerial image datasets.

The three datasets listed above are processed individually.

- VEDAI:
  - Calculate the width and height of GT based on its 4 corners:

$$w = x_{\max} - x_{\min}, h = y_{\max} - y_{\min}$$

  - Direction should be removed from annotations
  - Delete the annotation of “plane”, “boat”, and other three classes in labels
  - Label all four-wheeled vehicles(such as: truck, van,pickup,...) as "vehicle"
- VAID:
  - YOLO format should replace XML annotations
  - The "vehicle" label should be used for all types of cars
- DOTA:
  - Remove all labels except for "large vehicle" and "small vehicle". Both labels are combined to form one "car" label.
  - Divide the DOTA images into 1024 * 1024 squares. To ensure the positions of GT’s four corners in the new images are correct, the coordinates of each corner are converted accordingly.
  - Calculate center point coordinate and width and height:

$$x = (x_{\max} + x_{\min})/2, y = (y_{\max} + y_{\min})/2$$
$$w = x_{\max} - x_{\min}, h = y_{\max} - y_{\min}$$

Table 3 shows the information from the new datasets after they’ve been processed.

Dataset	Image Format	Images	Classes	Image size	Annotations
VEDAI	JPEG	1,250	1	512×512	$(class, x, y, w, h)$
VAID	JPEG	5,985	1	1137×640	$(class, x, y, w, h)$
DOTA	JPEG	14,348	1	1024×1024	$(class, x, y, w, h)$
Total	JPEG	21,583	1	$512 \times 512, 1024 \times 1024, 1137 \times 640$	$(class, x, y, w, h)$

Table 3: Information on the processed datasets

4 Object Tracking

Tracking objects involves using sensor measurements to determine their location, path, and characteristics.The term "sensor" can refer to any device capable of measuring objects in the physical environment, such as radar, sonar, ladar, cameras, infrared sensors, microphones, ultrasounds, etc. Tracking usually aims to determine the number, identity, and state of objects, including their positions, velocity, and in some cases their features.In the field of computer vision, object tracking is a crucial task. In recent years, the availability of high-powered computers, high-quality, and inexpensive video cameras, as well as the increasing interest in automated video analysis have led to an increased interest in object tracking algorithms.An object can be tracked as it moves around a scene in the image plane by estimating its trajectory. Another way to put it: A tracker labels the tracked objects consistently throughout multiple frames within a video. A tracker may also display object-centric information based on the tracking domain, such as orientation, area, or shape.Object tracking might be difficult due to the following factors[Yilmaz et al., 2006]:

- 2D image projected on a 3D world causes loss of information
- Image noise
- complex motion of objects
- non-rigidity of objects
- changing lighting conditions
- need for real-time processing

Visual Multiple Object Tracking (VMOT) method consists of mainly two parts: observation model and tracking process.The observation model as a descriptor of objects may be used to differentiate individuals, detect, and track objects by providing some traits that are unique to each. Tracking is a process for linking observations with trajectories. Most recent approaches aimed at solving the VMOT problem follow two main steps: object detection and data association. During detection, a pre-trained object detector is applied to identify some potential pixels in the input video. Following the discovery of the object candidates, the observation model represents them. The tracking process mostly involves forming connections between observation models and targets. There are two main classes of tracking methods: Bayesian theory based and data association based, depending on the way they link the observation to target.Our discussion focuses only on Bayesian theory based algorithms in this article.

4.1 Bayesian reasoning and object tracking

Bayesian reasoning is a well-developed theory that makes use of probabilistic and statistical techniques. This technique can be used to solve object tracking problems.Bayesian methods have become very popular in the engineering community. They've also been used in solving a wide variety of problems. In most tracking problems, sensor measurements are received in a sequential manner.During each measurement stage, the object state is re-estimated by combining new information with the current estimate. Bayes' theorem is a powerful framework for handling the sequential nature of data and the associated uncertainties. The Bayesian recursive approach uses Bayes' Theorem, the mathematical and conceptual basis upon which the Bayesian paradigm is built. On the basis of Bayesian filter theory, there are four tracking methods: Kalman filter, particle filter, Bayesian framework and RFSs Bayesian multiple target filtering-PHD.

4.1.1 Bayes' theorem

This theorem is the result of a philosophy that uses the conditional probability concepts of probability theory to consistently reconcile past and current information. It may be stated as follows: given two related events x and y, the conditional probability of event A given observation of event B is:

$$P(X \mid Y) = \frac{P(Y \mid X)P(X)}{P(Y)}$$

Let x be a random variable of interest. In object tracking, x often refers to the state of the object under consideration. Suppose y is a measurement related to x. Knowing y, we wish to update our understanding of x. The problem at the core of most tracking problems is how to update p(x) to take into account the new information y. The answer is by $P(X \mid Y)$, the conditional distribution of x given y.

Many industrial, engineering, economic, and other settings require repetitive measurements. Assume that at times $t_1, t_2, \ldots, t_k, \ldots$, we observe $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k, \ldots$ In object tracking, the y values are sensor measurements about the

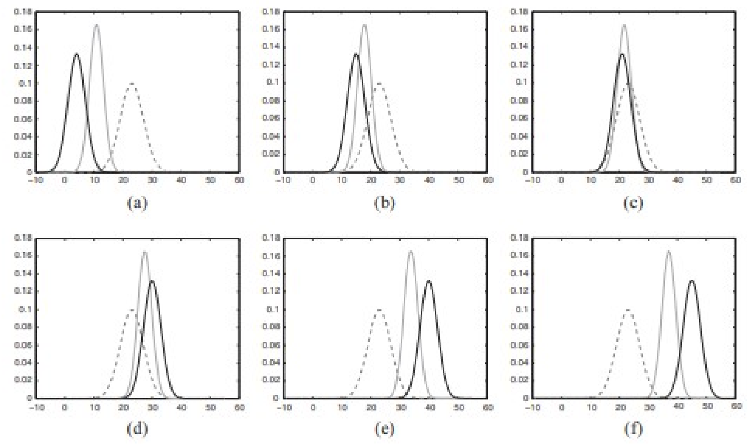


Figure 11: Prior p(x) (grey line), likelihood L(x) = p(ylx) (black line) and posterior p(xly) (grey line) for data values (a) y = 4, (b) y = 15, (c) y = 21, (d) y = 30, (e) y = 40, (f) y = 45

state of the object. Recursively treating the old posterior distribution as the new prior distribution, one value at a time, creates a recursive process.

$-$ at time 0 , before any measurement value arrives, $p(\mathbf{x})$ represents our knowledge of $\mathbf{x}$

- at time $t_1, p\left(\mathbf{x} \mid \mathbf{y}_1\right) \propto p\left(\mathbf{y}_1 \mid \mathbf{x}\right) p(\mathbf{x})$

- at time $t_2, p\left(\mathbf{x} \mid \mathbf{y}_1, \mathbf{y}_2\right) \propto p\left(\mathbf{y}_2 \mid \mathbf{y}_1, \mathbf{x}\right) p\left(\mathbf{x} \mid \mathbf{y}_1\right)$;

- at time $t_k, p\left(\mathbf{x} \mid \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{k-1}, \mathbf{y}_k\right) \propto p\left(\mathbf{y}_k \mid \mathbf{y}_{k-1}, \ldots, \mathbf{y}_1, \mathbf{x}\right) p\left(\mathbf{x} \mid \mathbf{y}_1, \ldots, \mathbf{y}_{k-1}\right)$.

4.2 Kalman filter

An algorithm known as Kalman filtering uses a series of observations over time to approximate unknown variables more accurately than those based solely on one measurement, by estimating a joint probability distribution over the variables over the given timeframe.[Kalman, 1960]. The Kalman filter was invented by mathematician and engineer Rudolf E. Kalman in the 1940s to solve problems in the fields of aerospace and control systems. Numerous technological applications exist for Kalman filtering. Guide, navigate, and control of dynamically positioned planes, spacecraft and ships is a common application. Kalman filtering is also a concept which is widely applied for time series analysis, including econometrics and signal processing. [Zarchan and Musoff, 2000] Kalman filtering is an efficient way to address multi-target tracking when the number of objects remains small. Due to the recursive nature of the method, identity switches become more frequent as the number of objects increases.[Berclaz et al., 2011]

4.3 DeepSORT: Deep Simple Real Time Tracker

the most popular and one of the most widely used, elegant object tracking framework is DeepSORT, which has been proposed by Nicolai Wojke, Alex Bewley and Dietrich Paulus in 2017.[Wojke et al., 2017a].

DeepSort consists of three main parts:

- Object Detection
- Track the possible path: This section is to deal with the occlusion and we used Kalman filter. see here
- Assign correct object

Suppose we want to track the position change of the car. As shown in figure 10. below, the blue distribution is the Kalman filter prediction value, the brown distribution is the sensor measurement value, and the gray distribution is the optimal estimation of the predicted value updated based on the measurement value.

The Kalman filter works by calculating a prediction function to determine how the measured quantities should change based on a model or a set of parameters. The filter then makes an optimal guess at those parameters based on past experience and other data that are assumed to be related directly or indirectly to those parameters. This means that it can

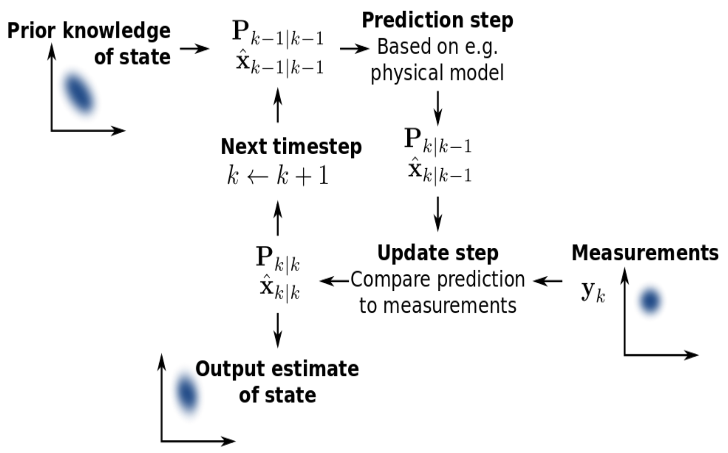
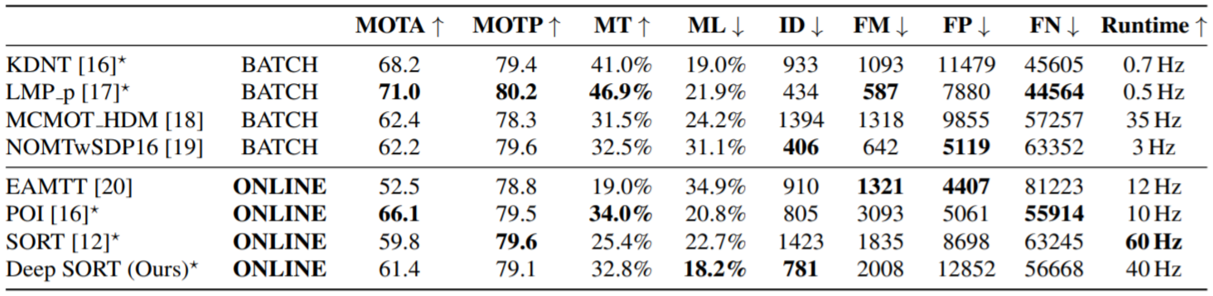


Figure 12: The Kalman filter keeps track of the estimated state of the system and the variance or uncertainty of the estimate. The estimate is updated using a state transition model and measurements. $\hat{x}_{k|k-1}$ denotes the estimate of the system's state at time step k before the k -th measurement y_k has been taken into account; $P_{k|k-1}$ is the corresponding uncertainty.



		MOTA ↑	**MOTP ↑**	**MT ↑**	**ML ↓**	**ID ↓**	**FM ↓**	**FP ↓**	**FN ↓**	**Runtime ↑**
KDNT [16]⋆	BATCH	68.2	79.4	41.0%	19.0%	933	1093	11479	45605	0.7 Hz
LMP_p [17]⋆	BATCH	**71.0**	**80.2**	**46.9%**	21.9%	434	**587**	7880	**44564**	0.5 Hz
MCMOT_HDM [18]	BATCH	62.4	78.3	31.5%	24.2%	1394	1318	9855	57257	35 Hz
NOMTwSDP16 [19]	BATCH	62.2	79.6	32.5%	31.1%	**406**	642	**5119**	63352	3 Hz
EAMTT [20]	**ONLINE**	52.5	78.8	19.0%	34.9%	910	**1321**	**4407**	81223	12 Hz
POI [16]⋆	**ONLINE**	**66.1**	79.5	**34.0%**	20.8%	805	3093	5061	**55914**	10 Hz
SORT [12]⋆	**ONLINE**	59.8	**79.6**	25.4%	22.7%	1423	1835	8698	63245	**60 Hz**
Deep SORT (Ours)⋆	**ONLINE**	61.4	79.1	32.8%	**18.2%**	**781**	2008	12852	56668	40 Hz

Figure 13: DeepSORT vs other tracker framework.[Wojke et al., 2017b]

take into account measurement-dependent noise in order for the resulting prediction error of any future measurement not to exceed some threshold value.

To show the current state of the object and predict its future state, we use the following vector:

where u and v represent the horizontal and vertical pixel location of the center of the target, while the scale s and r represent the scale (area) and the aspect ratio of the target's bounding box respectively

References

Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcrof. Simple online and realtime tracking. In *In 2016 IEEE International Conference on Image Processing (ICIP)*, pages 2–3. arXiv, 2017.

S. Redmon, J.and 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, Las Vegas*, pages 779–778. IEEE, 2016.

A. Redmon, J.and Farhadi. Yolo9000: Better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 21–26. IEEE, 2017a.

Ali Farhadi Joseph Redmon. Yolov3: An incremental improvement. In *updates to YOLO!*, pages 1–6. arXiv, 2018.

Trevor Darrell Jonathan Long, Evan Shelhamer. Fully convolutional networks for semantic segmentation. In *arXiv Computer Vision and Pattern Recognition*, pages 1–10. IEEE, 2015.

Junyan Lu, Chi Ma, Li Li, Xiaoyan Xing, Zhigang Wang Yong Zhan and, and Jiuwei Xu. A vehicle detection method for aerial image based on yolo. 2018.

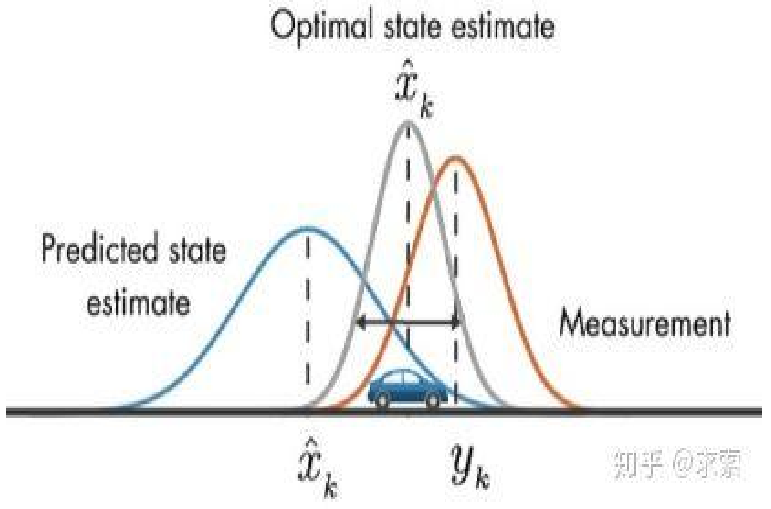


Figure 14: Kalman filter

$$\mathbf{x} = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]^T,$$

Figure 15: Two samples of the cropped images

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. arXiv, 2015.

A. Redmon, J.and Farhadi. Yolo9000: Better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2–4. IEEE, 2017b.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (2015)*. arXiv, 2015.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. arXiv, 2020.

Sebastien Razakarivony and Frederic Jurie. Vehicle detection in aerial imagery : A small target detection benchmark. *Journal of Visual Communication and Image Representation*, 34:187–203, 2016. ISSN 1047-3203. doi:https://doi.org/10.1016/j.jvcir.2015.11.002. URL `https://www.sciencedirect.com/science/article/pii/S1047320315002187`.

HUEI-YUNG LIN, KAI-CHUN TU, and CHIH-YI LI. Vaid: An aerial image dataset for vehicledetection and classification. IEEE, 2020.

Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images, 2019.

Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4): 13–es, 2006.

Rudolf E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 1960.

P. Zarchan and H. Musoff. *Fundamentals of Kalman Filtering: A Practical Approach*. Progress in astronautics and aeronautics. American Institute of Aeronautics and Astronautics, Incorporated, 2000. ISBN 9781563472541. URL `https://books.google.com/books?id=AQxRAAAAMAAJ`.

Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *IEEE transactions on pattern analysis and machine intelligence*, 33(9):1806–1819, 2011.

Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. arXiv, 2017a.

Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. page 4. arXiv, 2017b.