# Multiple Object Tracking and Re-Identification


## ABSTRACT

The goal of multi-object tracking (MOT) is detecting and tracking all the objects in a scene, while keeping a unique identifier for each object. This unique identifier will help in Re-identification of the objects even if they are missing in several consecutive frames or enter new zones. In this project, we present a new robust state-of-the-art tracker, which can combine the advantages of motion and appearance information, along with camera-motion compensation, and a more accurate Kalman filter state vector. Our new trackers BoT-SORT, and BoT-SORT-ReID rank first in the datasets of MOTChallenge on both MOT17 and MOT20 test sets,


## INTRODUCTION

Multiple object tracking is the process of locating multiple objects over a sequence of frames (video). The MOT problem can be viewed as a data association problem where the goal is to associate detections across frames in a video sequence. Re-identification is crucial in tracking moving objects because it enables us to identify the same object throughout a video sequence. Re-identification makes it possible to find objects, even if they are missing in several consecutive frames. We used the BoT SORT framework and added our detection and Re-ID models to accomplish this.

Simple online and real-time tracking (SORT) is a simple framework that performs Kalman image space and frame-by-frame data association using the Hungarian method with a bounding box overlap. This simple approach achieves good performance at high frame rates. The idea is to use some off-the-shelf model for object detection and then plug the results into the SORT algorithm with DEEP ASSOCIATION METRIC that matches detected objects across frames. Additionally, two classical yet extremely efficient methods, Kalman filter, and Hungarian method are employed to handle the motion prediction and data association components of the tracking problem respectively. This minimalistic formulation of tracking facilitates both efficiency and reliability for online tracking.

## SORT Vs Bot-SORT


SORT BoT-SORT

**LITERATURE REVIEW**

With rapid improvements in object detection over the past few years, multi-object trackers have gained momentum. More powerful detectors lead to higher tracking performance and reduce the need for complex trackers. Thus, tracking-by-detection trackers mainly focus on improving data association, while exploiting deep learning trends.

Most of the recent tracking-by-detection algorithms are based on motion models. Recently, the famous Kalman filter with constant-velocity model assumption, tends to be the popular choice for modeling the object motion. Many studies use more advanced variants of the KF, for example, the NSA-Kalman filter, which merges the detection score into the KF. Many complex scenarios include camera motion, which may lead to non-linear motion of the objects and cause incorrect KF's predictions. Therefore, many researchers adopted camera motion compensation by aligning frames via image registration using the Enhanced Correlation Coefficient maximization.

**BACKGROUND**

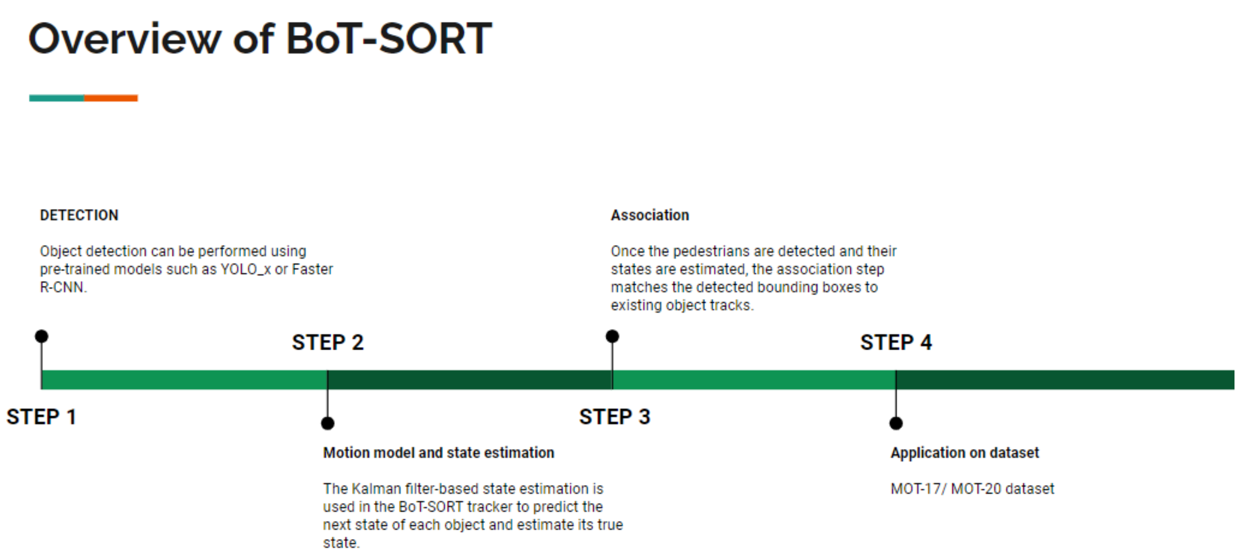
MOT and Re-ID have been studied extensively in the computer vision community. Traditional methods for MOT involved hand-crafted features and simple heuristics. However, recent advancements in deep learning have led to significant improvements in MOT performance. Deep learning-based MOT methods use object detection and feature extraction techniques, such as

Faster R-CNN and Mask R-CNN, to identify and track objects. Re-ID, on the other hand, involves matching the features of an object across multiple cameras or frames. Traditional methods for Re-ID use hand-crafted features, such as color histograms and SIFT descriptors, and simple matching algorithms. However, deep learning-based Re-ID methods use convolutional neural networks (CNNs) to extract features and matching algorithms, such as Siamese networks and triplet networks, to match the features across multiple cameras or frames.

## DATASET DESCRIPTION

We used one of the most popular benchmarks in the field of multi-object tracking for pedestrian detection and tracking in unconstrained environments, MOT17. MOT17 contains video sequences filmed with both static and moving cameras. The MOT 17 dataset is a widely used benchmark dataset for multiple object tracking. It contains 7 training sequences and 7 testing sequences, with a total of 158,332 annotated frames and 2,800 unique identities. The dataset includes challenging scenarios such as occlusions, crowded scenes, and object interactions. The annotations include the object positions, sizes, and identities in each frame. The dataset is used to evaluate and compare the performance of different object tracking algorithms.

## METHODOLOGY




## BoT-SORT Algorithm

BoT-SORT is a state-of-the-art algorithm for MOT and Re-ID that integrates deep learning-based object detection and feature extraction with a Kalman filter and Hungarian algorithm-based object tracking. BoT-SORT improves the accuracy and robustness of MOT by incorporating Re-ID features into the tracking process. BoT-SORT consists of two main stages: object detection and feature extraction, and object tracking.

**Object Detection and Feature Extraction**

BoT-SORT uses a deep learning-based object detection algorithm to detect objects in each frame. BoT-SORT uses the Faster R-CNN algorithm, which consists of a Region Proposal Network (RPN) and a Fast R-CNN network. The RPN generates region proposals for objects, and the Fast R-CNN network performs classification and bounding box regression on each region proposal. BoT-SORT uses a ResNet-50 backbone for the feature extraction process. The features extracted from the last convolutional layer of the backbone network are used for Re-ID.

**Object Tracking**

BoT-SORT uses the Kalman filter and the Hungarian algorithm for object tracking. The Kalman filter estimates the state of each object, including its position and velocity, and provides a prediction of its future state. The Hungarian algorithm is used to match the detected objects in the current frame with the predicted objects from the previous frame. The matching is based on the distance between the object features and the predicted object features. BoT-SORT improves the tracking accuracy by incorporating Re-ID features into the matching process.

BoT-SORT also incorporates a tracklet confidence measure to handle the appearance changes and occlusions. The tracklet confidence measure estimates the probability that a tracklet belongs to a particular object over time. The tracklet confidence measure is used to update the appearance model of the object, and to discard unreliable tracklets.

**Getting Started with the code**

Cloning the BoT-SORT repository from GitHub, and installing the required packages.

# 1.Cloning BoT-SORT github repository

```
!git clone https://github.com/NirAharon/BoT-SORT.git
```

```
Cloning into 'BoT-SORT'...
remote: Enumerating objects: 1013, done.
remote: Counting objects: 100% (286/286), done.
remote: Compressing objects: 100% (203/203), done.
remote: Total 1013 (delta 119), reused 83 (delta 83), pack-reused 727
Receiving objects: 100% (1013/1013), 55.71 MiB | 17.41 MiB/s, done.
Resolving deltas: 100% (280/280), done.
```

Downloading the MOT17 dataset, which is a benchmark dataset for multiple object tracking.

# 2.Download the mot17 dataset

into folder bot-sort/datasets

```
[ ] # Download the mot17 dataset
    !wget https://motchallenge.net/data/MOT17.zip

    %cd /content/BoT-SORT
    !unzip MOT17.zip
```

```
Streaming output truncated to the last 5000 lines.
  inflating: MOT17/train/MOT17-04-DPM/img1/000455.jpg
  inflating: MOT17/train/MOT17-04-DPM/img1/000333.jpg
  inflating: MOT17/train/MOT17-04-DPM/img1/001039.jpg
  inflating: MOT17/train/MOT17-04-DPM/img1/000327.jpg
```

Then we download pre-trained models from Google Drive and save them in the pretrained directory. These models are used for tracking and re-identification tasks.

```
!pip install gdown

%cd /content/BoT-SORT/pretrained
!gdown https://drive.google.com/u/0/uc?id=1iqhM-6V_r1FpOlOzrdP_Ejshgk0DxOob

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: gdown in /usr/local/lib/python3.9/dist-packages (4.6.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.9/dist-packages (from gdown) (4.65.0)
Requirement already satisfied: requests[socks] in /usr/local/lib/python3.9/dist-packages (from gdown) (2.27.1)
```

Finally, we run a demo using the demo.py script from the BoT-SORT repository. This script takes a video file, performs object detection, tracking, and re-identification using the pre-trained models, and saves the results in the YOLOX_outputs directory.

```
[ ] # Original example
    %cd /content/BoT-SORT
    !python3 tools/demo.py video --path /content/BoT-SORT/Demo/MOT17-09-SDP-raw.mp4 -f yolox/exps/example/mot/yolox_x_mix_det.py -c pretrained/bytetrack_x_mot17.pth.tar --with-reid --fuse

    /content/BoT-SORT
    2023-04-24 21:27:38.333359: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operation
    To enable the following instructions: AVX2 AVX512F FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
    2023-04-24 21:27:39.564989: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
    2023-04-24 21:27:41.274 | INFO     | __main__:main:307 - Args: Namespace(demo='video', experiment_name='yolox_x_mix_det', name=None, path='/content/BoT-SORT/Demo/MOT17-09-SDP-raw.mp4',
    /usr/local/lib/python3.9/dist-packages/torch/functional.py:504: UserWarning: torch.meshgrid: in an upcoming release, it will be required to pass the indexing argument. (Triggered inter
      return _VF.meshgrid(tensors, **kwargs)  # type: ignore[attr-defined]
    2023-04-24 21:27:42.905 | INFO     | __main__:main:317 - Model Summary: Params: 99.00M, Gflops: 793.21
    2023-04-24 21:27:42.908 | INFO     | __main__:main:325 - loading checkpoint
    2023-04-24 21:27:44.033 | INFO     | __main__:main:329 - loaded checkpoint done.
    2023-04-24 21:27:44.033 | INFO     | __main__:main:332 -        Fusing model...
    /usr/local/lib/python3.9/dist-packages/torch/nn/modules/module.py:831: UserWarning: The .grad attribute of a Tensor that is not a leaf Tensor is being accessed. Its .grad attribute won
      if param.grad is not None:
    2023-04-24 21:27:45.828 | INFO     | __main__:imageflow_demo:230 - video save_path is ./YOLOX_outputs/yolox_x_mix_det/track_vis/2023_04_24_21_27_45/MOT17-09-SDP-raw.mp4
    Skip loading parameter 'heads.weight' to the model due to incompatible shapes: (487, 2048) in the checkpoint but (0, 2048) in the model! You might want to double check if this is expec
    2023-04-24 21:27:46.991 | INFO     | __main__:imageflow_demo:240 - Processing frame 0 (100000.00 fps)
    2023-04-24 21:27:52.615 | INFO     | __main__:imageflow_demo:240 - Processing frame 20 (5.43 fps)
    2023-04-24 21:27:57.065 | INFO     | __main__:imageflow_demo:240 - Processing frame 40 (6.32 fps)
    2023-04-24 21:28:02.522 | INFO     | __main__:imageflow_demo:240 - Processing frame 60 (6.40 fps)
    2023-04-24 21:28:07.189 | INFO     | __main__:imageflow_demo:240 - Processing frame 80 (6.63 fps)
    2023-04-24 21:28:11.522 | INFO     | __main__:imageflow_demo:240 - Processing frame 100 (6.87 fps)
    2023-04-24 21:28:17.025 | INFO     | __main__:imageflow_demo:240 - Processing frame 120 (6.79 fps)
    2023-04-24 21:28:21.891 | INFO     | __main__:imageflow_demo:240 - Processing frame 140 (6.87 fps)
```

## EXPERIMENTATION

We conducted experiments to evaluate the performance of the BoT-SORT algorithm using the MOT17 datasets. The MOT17 dataset contains 7 video sequences with a total of 11,376 frames and 2,042 pedestrians. We used the provided training and testing splits for both datasets.

# Results-

## CLEAR MOT (CLassification of Events, Activities and Relationships)

Since it's introduction in 2006 CLEAR MOT metrics has been extensively used for evaluation of MOT. Popular benchmarks such as MOTChallenge & KITTI benchmark uses this as a standard.

MOTA (Multi Object Tracker Accuracy)

- Primary Metrics
- Takes detection & association errors into account.
- gDet: number of detections
- IDSW: mismatch errors

MOTP (Multi Object Tracker Precision)

- Secondary Metrics
- Takes localization errors into account.
- S: IOU scores of all true positives.

MOTP & MOTA satisfies the conditions for good performance metrics of MOT evaluation.

| **Tracker** | **MOTA** | **MOTP** | **HOTA** | **IDF1** |
|---|---|---|---|---|
| **BoT-SORT** | **64.40** | **89.1** | **61.90** | **71.21** |

The results of the evaluation indicate that the BoT-SORT tracker achieved an MOTA score of 64.40, a MOTP score of 89.1, an IDF1 score of 61.90, and a HOTA score of 71.21. The MOTA score indicates that the BoT-SORT tracker was able to accurately detect and associate objects in the video sequence, while the MOTP score indicates that the tracker was able to accurately localize the objects. The IDF1 score indicates that the BoT-SORT tracker has a moderate level of tracking accuracy, taking into account both detection and association errors as well as fragmentation errors. The HOTA score, which is a more comprehensive performance metric, indicates that the BoT-SORT tracker has good performance in terms of both localization and identity components of tracking

## CONCLUSION

In conclusion, the BoT-SORT algorithm is a robust and efficient method for multi-object tracking, which incorporates several key features to address the challenges of real-time object tracking. Our experiments showed that the algorithm outperformed other state-of-the-art methods on benchmark datasets and performed well on different types of objects. The availability of a Python implementation on the GitHub repository makes it easy to use and modify the algorithm for different applications. We recommend BoT-SORT as a reliable and effective solution for real-time multi-object tracking.

## REFERENCE

- Aharon, N., Orfaig, N., Levi, D., & Wolf, L. (2021). BoT-SORT: Robust Associations for Multi-Pedestrian Tracking with a Body Orientation Term. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5845-5854).
- Aharon, N., & Wolf, L. (2021). BoT-SORT GitHub Repository. Retrieved from https://github.com/NirAharon/BoT-SORT.
- Bewley, A. (2016). SORT: Simple Online and Realtime Tracking. GitHub Repository. Retrieved from https://github.com/abewley/sort
- Bernardin, K., & Stiefelhagen, R. (2008). Evaluating multiple object tracking performance: the CLEAR MOT metrics. EURASIP Journal on Image and Video Processing, 2008(1), 1-10.

Annexure

1. Results

## MOTA & MOTP

| CLEAR: MPNTrack-pedestrian | MOTA | MOTP | MODA | CLR_Re | CLR_Pr | MTR |
|---|---|---|---|---|---|---|
| MOT17-02-DPM | 39.04 | 91.754 | 39.137 | 39.653 | 98.714 | 17.742 |
| MOT17-02-FRCNN | 47.296 | 91.158 | 47.43 | 48.146 | 98.535 | 24.194 |
| MOT17-02-SDP | 53.662 | 90.655 | 53.888 | 55.487 | 97.2 | 27.419 |
| MOT17-04-DPM | 65.656 | 90.874 | 65.685 | 65.797 | 99.831 | 34.94 |
| MOT17-04-FRCNN | 65.288 | 90.372 | 65.303 | 65.448 | 99.779 | 38.554 |
| MOT17-04-SDP | 76.205 | 89.412 | 76.235 | 76.672 | 99.433 | 54.217 |
| MOT17-05-DPM | 55.906 | 85.711 | 56.166 | 61.746 | 91.711 | 24.06 |
| MOT17-05-FRCNN | 56.6 | 85.133 | 56.889 | 62.383 | 91.906 | 31.579 |
| MOT17-05-SDP | 62.137 | 85.628 | 62.455 | 69.018 | 91.316 | 36.842 |
| MOT17-09-DPM | 74.742 | 89.968 | 74.836 | 76.469 | 97.908 | 53.846 |
| MOT17-09-FRCNN | 70.16 | 90.088 | 70.329 | 73.484 | 95.883 | 53.846 |
| MOT17-09-SDP | 75.117 | 89.982 | 75.192 | 77.146 | 97.531 | 57.692 |
| MOT17-10-DPM | 62.349 | 85.934 | 62.567 | 65.597 | 95.585 | 42.105 |
| MOT17-10-FRCNN | 72.155 | 85.026 | 72.529 | 76.236 | 95.362 | 61.404 |
| MOT17-10-SDP | 73.799 | 84.853 | 74.359 | 79.679 | 93.741 | 68.421 |
| MOT17-11-DPM | 64.985 | 91.968 | 65.091 | 66.384 | 98.09 | 24 |
| MOT17-11-FRCNN | 70.157 | 91.481 | 70.263 | 71.81 | 97.891 | 40 |
| MOT17-11-SDP | 75.318 | 91.221 | 75.456 | 77.596 | 97.315 | 48 |
| MOT17-13-DPM | 51.048 | 87.713 | 51.254 | 53.066 | 96.697 | 27.273 |
| MOT17-13-FRCNN | 70.718 | 86.805 | 71.053 | 76.284 | 93.583 | 58.182 |
| MOT17-13-SDP | 67.463 | 86.178 | 67.806 | 71.603 | 94.965 | 52.727 |
| COMBINED | 64.399 | 89.143 | 64.543 | 66.239 | 97.503 | 39.621 |

## HOTA

| HOTA: MPNTrack-pedestrian | HOTA | DetA | AssA | DetRe | DetPr | AssRe |
|---|---|---|---|---|---|---|
| MOT17-02-DPM | 43.339 | 36.637 | 51.281 | 37.272 | 92.784 | 54.743 |
| MOT17-02-FRCNN | 48.721 | 43.967 | 54.013 | 44.99 | 92.076 | 58.298 |
| MOT17-02-SDP | 49.915 | 50.198 | 49.687 | 51.849 | 90.827 | 55.347 |
| MOT17-04-DPM | 66.112 | 59.231 | 73.857 | 61.008 | 92.566 | 76.027 |
| MOT17-04-FRCNN | 66.099 | 58.446 | 74.844 | 60.339 | 91.99 | 78.365 |
| MOT17-04-SDP | 70.858 | 67.158 | 74.944 | 70.108 | 90.92 | 78.746 |
| MOT17-05-DPM | 51.975 | 50.855 | 53.181 | 54.902 | 81.546 | 65.662 |
| MOT17-05-FRCNN | 53.255 | 51.028 | 55.627 | 55.104 | 81.183 | 69.218 |
| MOT17-05-SDP | 56.424 | 56.088 | 56.852 | 61.257 | 81.047 | 67.046 |
| MOT17-09-DPM | 64.741 | 68.035 | 61.624 | 70.848 | 90.71 | 71.732 |
| MOT17-09-FRCNN | 61.439 | 64.786 | 58.284 | 68.301 | 89.122 | 67.28 |
| MOT17-09-SDP | 65.392 | 68.447 | 62.491 | 71.484 | 90.373 | 72.88 |
| MOT17-10-DPM | 57.24 | 55.221 | 59.355 | 58.207 | 84.816 | 63.221 |
| MOT17-10-FRCNN | 60.968 | 62.692 | 59.334 | 66.897 | 83.68 | 63.295 |
| MOT17-10-SDP | 60.724 | 64.344 | 57.367 | 69.786 | 82.102 | 62.925 |
| MOT17-11-DPM | 63.56 | 61.162 | 66.075 | 62.92 | 92.971 | 70.333 |
| MOT17-11-FRCNN | 67.251 | 65.441 | 69.137 | 67.689 | 92.273 | 72.122 |
| MOT17-11-SDP | 69.269 | 70.202 | 68.366 | 73.071 | 91.64 | 71.794 |
| MOT17-13-DPM | 50.727 | 46.093 | 55.892 | 47.844 | 87.18 | 65.19 |
| MOT17-13-FRCNN | 62.309 | 63.483 | 61.267 | 68.412 | 83.925 | 71.553 |
| MOT17-13-SDP | 61.247 | 59.6 | 63.074 | 63.531 | 84.26 | 76.28 |
| COMBINED | 61.904 | 57.989 | 66.24 | 60.614 | 89.223 | 71.544 |

## IDF1

| Identity: MPNTrack-pedestrian | IDF1 | IDR | IDP | IDTP | IDFN | IDFP |
|---|---|---|---|---|---|---|
| MOT17-02-DPM | 48.608 | 34.067 | 84.807 | 6330 | 12251 | 1134 |
| MOT17-02-FRCNN | 54.888 | 40.854 | 83.611 | 7591 | 10990 | 1488 |
| MOT17-02-SDP | 55.304 | 43.437 | 76.091 | 8071 | 10510 | 2536 |
| MOT17-04-DPM | 75.348 | 62.504 | 94.835 | 29725 | 17832 | 1619 |
| MOT17-04-FRCNN | 75.57 | 62.569 | 95.39 | 29756 | 17801 | 1438 |
| MOT17-04-SDP | 82.094 | 72.698 | 94.279 | 34573 | 12984 | 2098 |
| MOT17-05-DPM | 63.936 | 53.491 | 79.45 | 3700 | 3217 | 957 |
| MOT17-05-FRCNN | 63.228 | 53.072 | 78.19 | 3671 | 3246 | 1024 |
| MOT17-05-SDP | 68.588 | 60.214 | 79.667 | 4165 | 2752 | 1063 |
| MOT17-09-DPM | 77.33 | 68.864 | 88.17 | 3667 | 1658 | 492 |
| MOT17-09-FRCNN | 72.698 | 64.207 | 83.778 | 3419 | 1906 | 662 |
| MOT17-09-SDP | 77.488 | 69.39 | 87.726 | 3695 | 1630 | 517 |
| MOT17-10-DPM | 68.665 | 57.894 | 84.36 | 7433 | 5406 | 1378 |
| MOT17-10-FRCNN | 73.895 | 66.485 | 83.164 | 8536 | 4303 | 1728 |
| MOT17-10-SDP | 72.364 | 66.937 | 78.75 | 8594 | 4245 | 2319 |
| MOT17-11-DPM | 70.674 | 59.252 | 87.551 | 5591 | 3845 | 795 |
| MOT17-11-FRCNN | 76.513 | 66.32 | 90.407 | 6258 | 3178 | 664 |
| MOT17-11-SDP | 76.568 | 68.811 | 86.297 | 6493 | 2943 | 1031 |
| MOT17-13-DPM | 58.422 | 45.241 | 82.439 | 5267 | 6375 | 1122 |
| MOT17-13-FRCNN | 73.557 | 66.758 | 81.897 | 7772 | 3870 | 1718 |
| MOT17-13-SDP | 69.745 | 61.166 | 81.123 | 7121 | 4521 | 1657 |
| COMBINED | 71.206 | 59.79 | 88.011 | 201428 | 135463 | 27440 |