# CV3DST Multi-Object Tracking Challenge: Tracking with Bounding Box Regression

Orcun Cetintas
Technical University of Munich
orcun.cetintas@tum.de

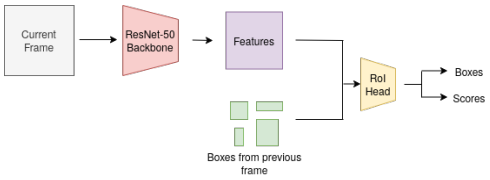


Figure 1. Tracking with regression of the bounding boxes of the previous frame


Figure 2. Detection of the new tracks with the two-stage detector

## 1. Introduction

The goal in multi-object tracking (MOT) is to find the trajectories of the objects of interest in a video. MOT is a challenging task due to heavy occlusions, multiple objects of the same type, and similar appearances of the objects in the frame.

Tracktor [1] utilizes the two-stage object detectors' bounding box regression capability to perform MOT and achieves state-of-the-art results. The success of the Tracktor motivated me to perform tracking with bounding box regression for the CV3DST Challenge. I used the baseline tracker which is provided by the CV3DST team as the starting code. Similarly, I used the pre-trained Faster R-CNN [4] network with ResNet-50 [2] backbone which is also provided by the team as the object detector. Aside from the provided files, all the improvements are implemented by me, and the code is available at `https://github.com/ocetintas/tracktor-mot`

## 2. Methodology

**Tracking by bounding box regression [1]** Two-stage detectors are capable of regressing the bounding boxes of the proposals. Under the assumption of the slow movement of the objects, one can utilize this property to track the objects detected in the previous frame as illustrated in Figure 1. If a new object appears in the current frame, the object detection capability of the two-stage detector can be used to localize the object, as illustrated in Figure 2. Finally, one can check the IoU between the detected boxes and regressed boxes to decide to initialize a new track. I implemented the bounding box regression, data association, and new track detection modules by following the Tracktor paper [1]. The first version of my implementation was not using NMS, and this version will be referred to as *t-naive*. The second version performs NMS after both detection and regression of the bounding boxes and it will be referred to as *t-basic*.

**Trajectory prediction** t-basic uses previous frame's bounding boxes and in the cases where the objects move fast, it may lose the track. Movement of the objects can be compensated by assuming they move with constant velocity. With this assumption, I implemented a trajectory prediction module that shifts the bottom-left corner of the bounding boxes before performing box regression, and this version will be referred to as *t-trajectory*.

Even though t-trajectory achieves good results, there are few problems with this assumption, specifically in the cases where the camera changes its direction, pedestrians slow down/start moving and occlude each other. Therefore, I decided to not solely depend on constant velocity assumption. Instead, I used boxes coming from trajectory prediction and the previous frame's bounding boxes as object proposals. I fed all the candidate proposals to the RoI Head of the object detector and chose only the best scores among both categories. Hence, this version uses the best of both worlds and will be referred to as *t-proposal*.

**Hyperparameters and model comparison** Tracking with bounding box regression requires a hyperparameter that controls the NMS threshold after bounding box regression. This parameter is specifically important for the scenarios in which multiple pedestrians are walking together

| $\lambda_{active}$ | MOTA↑ | IDF1↑ | IDSW ↓ | MT↑ | ML↓ |
|---|---|---|---|---|---|
| 0.3 | 67.3 | 57.8 | 3488 | 268 | 58 |
| 0.45 | 69.3 | 62.3 | 2984 | 283 | 57 |
| 0.6 | **69.6** | **63.8** | **2964** | **285** | 57 |
| 0.75 | **69.6** | 63.3 | 2983 | 284 | **56** |

Table 1. Hyperparameter search on $\lambda_{active}$ with t-basic

| Method | MOTA↑ | IDF1↑ | IDSW ↓ | MT↑ | ML↓ |
|---|---|---|---|---|---|
| t-naive | 57.1 | 60.0 | 2880 | **295** | **54** |
| t-basic | **69.6** | 63.8 | 2964 | 285 | 57 |
| t-trajectory | **69.6** | 64.3 | 2400 | 286 | 56 |
| t-proposal | **69.6** | **64.9** | **2246** | 288 | 56 |

Table 2. Performance of the tracker versions

or occluding each other. I initially set this parameter to 0.3 which is a common threshold for NMS. However, this resulted in a high number of ID switches and low IDF1 score. Bergmann et al. [1] refer to this parameter as $\lambda_{active}$ and set it to 0.6. I ran a hyperparameter search on this parameter as illustrated in Table 1. Choosing $\lambda_{active} = 0.75$ and $\lambda_{active} = 0.6$ improved the tracker performance significantly.

Table 2 presents the performance of the different versions of the tracker on the MOT16 [3] training set. t-proposal is the most successful version in MOTA and IDF1 scores which are the main objectives of the CV3DST challenge. Moreover, t-basic and t-trajectory achieve really good results by obtaining the same MOTA score with t-proposal even though they have slightly lower IDF1 scores.

## 3. Challenge Results

After cross-validation on the training set, I submitted t-basic and t-trajectory for evaluation on the test set. Table 3 shows the best performing results of the CV3DST challenge. Both t-basic and t-trajectory outperforms other submissions and achieves the best MOTA score. Even though t-trajectory performed better on the training set, t-basic achieved a slightly higher MOTA and IDF1 scores on the test set. On the other hand, both versions have a relatively lower IDF1 score. This indicates that in difficult scenarios the model is struggling to associate identities, as it is not using any re-identification strategy.

As the allowed submission number was limited and I was already above the threshold; I wanted to save my last right to submit in case something goes wrong. Hence, I decided to not evaluate t-proposal on the test set. However, t-proposal might achieve slightly better IDF1 scores than both t-basic and t-trajectory as cross-validation results indicate.

| Method | MOTA↑ | IDF1↑ | IDSW ↓ | MT↑ | ML↓ |
|---|---|---|---|---|---|
| **t-basic** | 67.80 | 56.74 | 870 | 128 | 74 |
| **t-trajectory** | 67.35 | 55.87 | 938 | 126 | 74 |
| s1063 | 65.76 | 60.66 | 600 | 120 | 72 |
| s1066 | 65.61 | 54.65 | 639 | 118 | 78 |

Table 3. CV3DST tracking challenge best performing results

## References


[1] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixé. Tracking without bells and whistles. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 941–951. IEEE, 2019.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[3] Anton Milan, Laura Leal-Taixé, Ian D. Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *CoRR*, abs/1603.00831, 2016.

[4] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99, 2015.