Joint detection and online multi-object tracking

Hilke Kieritz, Wolfgang Hübner, and Michael Arens
Fraunhofer IOSB, Germany
{hilke.kieritz|wolfgang.huebner|michael.arens}@iosb.fraunhofer.de

Abstract

Most multiple object tracking methods rely on object detection methods in order to initialize new tracks and to update existing tracks. Although strongly interconnected, tracking and detection are usually addressed as separate building blocks. However both parts can benefit from each other, e.g. *the affinity model from the tracking method can reuse appearance features already calculated by the detector, and the detector can use object information from past in order to avoid missed detection.*
Towards this end, we propose a multiple object tracking method that jointly performs detection and tracking in a single neural network architecture. By training both parts together, we can use optimized parameters instead of heuristic decisions over the track lifetime. We adapt the Single Shot MultiBox Detector (SSD)[14] to serve single frame detection to a recurrent neural network (RNN), which combines detections into tracks. We show initial prove of concept on the DETRAC[26] benchmark with competitive results, illustrating the feasibility of learnable track management. We conclude with a discussion of open problems on the MOT16[15] benchmark.

1. Introduction

The main challenge in tracking multiple objects in a video sequence stems from its open world nature. Specifically, the unknown number of present objects as well as uncertainties and ambiguities in assigning tracks to detection (data association) make it difficult to decide if a detection is a false positive or a new track, or when a track should be established or terminated. Even more challenging is tracking objects in an online setting, where a globally optimal solution cannot be achieved. Locality in time means that once an error is made it can hardly be corrected in the future, *e.g.*, confusing a false positive detection on a background structure with a new track could result in a false track, which will constantly be confirmed by the background structure.
Multiple object tracking methods typical consist of the following components: a detection method, a detection-track affinity metric, and a track management. The detection method is an object detector finding bounding boxes enclosing instances specific object categories. The affinity metric based on the track representation is used to combine detections into tracks. The track management includes decisions of when to create new tracks and when to terminate old tracks. Further, it decides how to update a track, *e.g.* if an associated detection can be trusted.
Leal-Taixe *et al.* [13] conclude that a strong affinity model, often based on appearance cues, is important for good tracking performance and that deeply learned models show the most potential. Many current approaches for tracking multiple objects view the detection method and the affinity metric as completely independent components. However, it can be assumed that both components are strongly related, and that utilizing this relation can lead to an improved system performance [11]. Further, can the detection method benefit from information of the tracking components. So could the non-maximum suppression of the detector benefit from knowing the last position of the tracks, preventing the generation of false negative errors.
The track management is responsible for keeping the balance between connecting detections to tracks and discarding unlikely matches. Its parameters are dependent on detector and affinity metric performance and are often set by hand-tuned heuristics, *e.g.* splitting detections into strong and weak ones [20, 11], or the number of frames until a lost track is terminated [27, 11, 7]. Tuning these parameters manually for a given setting is tedious task, which in most cases leads to sub-optimal solutions.
Towards this end, we propose an online multiple object tracking method that integrates all components in a single neural network. We use a recursive neural network (RNN) to associate the tracks with the detections and update the track representation in each frame. We propose a track-detection affinity metric based on spatial distance, object appearance, detection score and track score. Our appearance clues are built on features that are used by the detector. Since it is easier to create a comprehensive training set for detection than for tracking, we train our method in two phases. First, we train the detector with a large detection

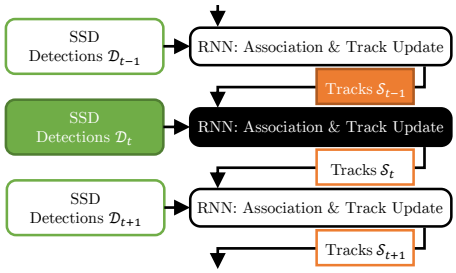


Figure 1: This figure depicts the unrolled RNN and shows the integrations of the SSD.

training set, than we add the RNN and train on a significantly smaller amount of tracking sequences.

2. Related Work

In general, one can differentiate between offline methods, which globally optimize the tracks, and online methods, which base their decisions on previous observed data. Many offline methods are formulated as a graph optimization problem, *e.g*. minimax path[21], or minimum cost multicut[24, 25], with the detection as the nodes and the affinity or distance metric for the weights of the edges. Online methods are using an affinity metric to connect tracks and detections in each frame. Often the Hungarian algorithm is applied to find optimal associations[18, 5, 29]. Recently several multiple object tracking methods include deeply learned features in their affinity metric[2, 7, 18, 21], in which the features are calculated independent of the object detector. In [16] Milan *et al*. present an end-to-end trained recurrent neural network for online tracking. Similar to our proposed method it combines detections into tracks over time using an RNN, but contrary to our proposed method, it does not utilized appearance features and does not include the detection method in to their network.
Another set of methods focus on detection of multiple object categories in videos[23, 10] by utilizing deep learning. But the goal of these methods is to increase the detection performance and not to preserve a consistent instance ID over multiple frames.

3. Joint Detection and Multiple Object Tracking Approach

Our method builds onto the Single Shot MultiBox Detector (SSD)[14] pipeline. The SSD detector processes an image along a fixed set of bounding boxes ("prior boxes"). For each prior box, classification scores are calculated and a regressor is used to calculate a refined object localization. We adapt the network to additionally output appearance features $\mathcal{A}_t$ for each prior box by adding an additional convolution layer to every output layer of the SSD. This is the input for the recurrent neural network (RNN) (see figure 1), which is responsible for combining the tracks from the last frame with the current detections. Therefore, it calculates an association metric (section 3.2) between the tracks and the detections. This metric is used to guide a modified non-maximum suppression (NMS), which replaces the NMS used in the SSD. The NMS is integrated in the network as a selection layer (section 3.3). The association metric is further used to associate detections to tracks by solving a bipartite graph matching with the Hungarian algorithm (section 3.4), which is integrated as a permutation layer. Further, a track score is calculated (section 3.5) based on the old track confidence, detection confidence, and association confidence. We use a fixed number of tracks $\mathcal{S}_t$, where old tracks are replaced with new ones based on their track score (see section 3.6). Each track consist of a track ID ψ, a track score l and a history of Δ_{max} matched detections $q_1, \ldots, q_{\Delta_{max}}$ with a flag $\xi_1, \ldots, \xi_{\Delta_{max}}$ that indicates if a matched detection is valid. See figure 2 for an overview of the recurrent network.

3.1. Detection

In each frame t we use the SSD to generate a set of detections $\mathcal{D}_t^{SSD} = \left\{\left(c_t^{bg}, c_t^1, \ldots, c_t^{max}, x'_t, y'_t, w'_t, h'_t\right)\right\}$, with c_t^i the classification scores and (x'_t, y'_t, w'_t, h'_t) the bounding box regression. For each detection we calculate a single detection score c_t^d based on the class predictions: $c_t^d = \max_i c_t^i - c^{bg}$. The detection center position (x_t^d, y_t^d) and size (w_t^d, h_t^d) are calculated from the prior box x^p, y^p, w^p, h^p and the estimated box regression: $x_t^d = x^p + x'_t$ etc. In combination with the appearance features $\mathcal{A}_t^d$ for each detection this yields the input detection set $\mathcal{D}_t = \left\{q_t^d\right\}$ for the recurrent network, with $q_t^d = \left(c_t^d, x_t^d, y_t^d, w_t^d, h_t^d, \mathcal{A}_t^d\right)$. Since no non-maximum suppression has taken place yet, the number of detections $|\mathcal{D}_t|$ in each frame is fixed depending on the number of prior-boxes used in the SSD architecture.

3.2. Affinity Measure

Before filtering detections with a low score by the non-maximum suppression, we calculate an affinity score between each detection in the input detection set $\mathcal{D}_t$ and each track of the previous frame $\mathcal{S}_{t-1}$. For that we observe the history of associations between detection and tracks for the last Δ_{max} frames. We keep the history, because a track could not be visible in the last frame, or could have a non-representative appearance, when it is half occluded. Therefore, for each time step Δ in the track history we calculate the euclidean distance $a_{s,d,\Delta}$ between the appearance

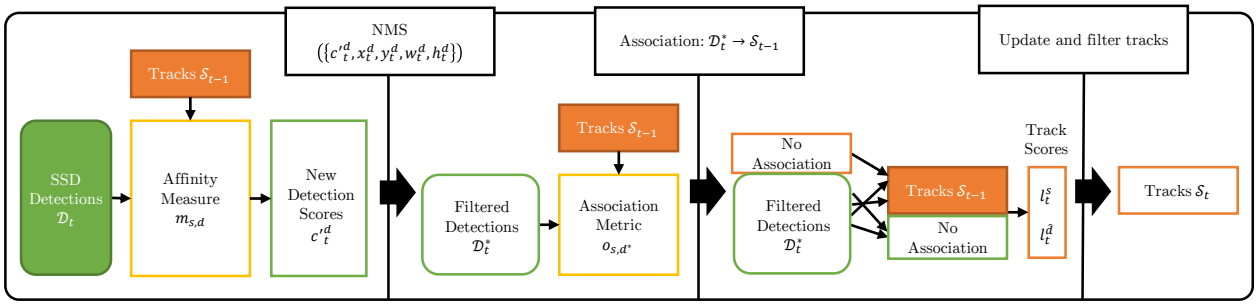


Figure 2: Overview of RNN operations. From detections $\mathcal{D}_t$ and tracks $\mathcal{S}_{t-1}$ to updated tracks $\mathcal{S}_t$.

of the detection $\mathcal{A}^d_t$ and the appearance of the track given by the associated detections $\mathcal{A}^s_\Delta$. Further the position distance is given by the euclidean distance $b_{s,d,\Delta}$ between bounding box centers $\left(x^d_t, y^d_t\right)$ and (x^s_Δ, y^s_Δ) and the euclidean distance $f_{s,d,\Delta}$ between bounding box size $\left(w^d_t, h^d_t\right)$ and (w^s_Δ, h^s_Δ). The position distances are adapted to include a scale factor depending on the size of the detection, *i.e.* the same distance for a small detection should result in a greater error than for a large detection: $b'_{s,d,\Delta} = r^d_t \cdot r^s_\Delta \cdot b_{s,d,\Delta}$ and $f'_{s,d,\Delta} = r^d_t \cdot r^s_\Delta \cdot f_{s,d,\Delta}$, with $r^d_t = R\left(w^d_t, h^d_t\right)$ and $r^s_\Delta = R\left(w^s_\Delta, h^s_\Delta\right)$. $R(\cdot)$ is calculated with a small multilayer perceptron with one hidden layer. From these distances we calculate the affinity metric between each detection d and each time step Δ in the history of each track s:

$$m_{s,d,\Delta} = M_\Delta\left(a_{s,d,\Delta}, b'_{s,d,\Delta}, f'_{s,d,\Delta}, c^d_t\right),$$

with $M_\Delta(\cdot)$ being a multilayer perceptron with two hidden layers and c^d_t the detection score. The affinity between a detection and a track is then given by the maximum of all valid detections of the track history:

$$m_{s,d} = \begin{cases} \max_\Delta \left\{m_{s,d,\Delta} \mid \xi^s_\Delta = 1\right\}, & \text{if } \sum_\Delta \xi^s_\Delta > 0 \\ m_\tau, & \text{otherwise} \end{cases}.$$

If the track has no valid associated detection, the track itself is regarded as invalid and a dummy value m_τ is used as affinity measure.

3.3. Non-Maximum Suppression

We use a greedy non-maximum suppression method, which first filters detections with small confidence and then removes all detections that are overlapped by another detection with a higher confidence. Because we do not use a position prediction for the tracks, the tracks depend on an associated detection in order to be updated. To avoid that detections that belong to a track with low detection scores are filtered out by the rough non-maximum suppression, the detection scores are re-weighted using the maximum similarity $m_d = \max_s\left(m_{s,d}\right)$ to the tracks. The new detection score is given by $c'^d_t = C(c^d_t, m_d)$, with $C(\cdot)$ being a multilayer perceptron with two hidden layers. The non-maximum suppression is implemented as a selection layer, which yields a subset of detections $\mathcal{D}^*_t = NMS\left(\left\{\left(c'^d_t, x^d_t, y^d_t, w^d_t, h^d_t\right)\right\}\right) \subset \mathcal{D}_t$. The size of the subset $|\mathcal{D}^*_t|$ depends on the non-maximum suppression, but is limited by the NMS parameter D^*.

3.4. Track-Detection Association

We calculate the association between the tracks $\mathcal{S}_{t-1}$ and the filtered detections $\mathcal{D}^*_t$ by solving a bipartite graph matching using the Hungarian algorithm. Besides the affinity measure m_{s,d^*}, we use the track l^s_{t-1} and the detection score $c^{d^*}_t$ for the association metric:

$$o_{s,d^*} = O\left(l^s_{t-1}, c^{d^*}_t, m_{s,d^*}\right).$$

This gives the method the ability to prefer high scoring tracks to lower scoring ones, if a conflict occurs. $O(\cdot)$ is a multilayer perceptron with two hidden layers. In most cases, there are more detection than tracks to associate. In addition, if a pair has an association value smaller than zero, the association is discarded. Further q^s_t is the detection matched to track s and $q^s_t = \tau$ if the track has no match. The set of detections without an associated track $\hat{q}$ is $\hat{\mathcal{D}}_t = \mathcal{D}^*_t \setminus \{q^s_t | s\}$.

3.5. Track Score

The next step is to calculate a new track score l^s_t for all existing tracks and for all potentially new tracks, consisting of detections without an associated track $\hat{\mathcal{D}}_t$. The track score is based on the last track score l^s_{t-1}, the detection score of the associated detection $c^{d^s}_t$, and the association measure m_{s,d^s}:

$$l^s_t = L\left(l^s_{t-1}, c^{d^s}_t, m_{s,d^s}, a_{s,d^s}\right),$$

with $L(\cdot)$ being a small multilayer perceptron with two hidden layers. If a track has no association, dummy values are

learned for the missing detection score c^τ and association measure m_τ. Similar a dummy value l^τ is learned for detection without associated tracks.

3.6. Track Management

The last step is to manage the track life cycle, *i.e.* to start new tracks, to conclude old ones and to update existing tracks. We do not explicitly terminate tracks but keep a limited number of tracks in the set. Therefore we rank the new and old tracks by their track score l_t^s and keep the highest scoring tracks. For updating the tracks, the associated detection is included in the history of each track: $q'^s_{\Delta_{max}} = q^s_{\Delta_{max}-1}, \ldots, q'^s_2 = q^s_1$ and $\xi'^s_{\Delta_{max}} = \xi^s_{\Delta_{max}-1}, \ldots, \xi'^s_2 = \xi^s_1$, with

$$q'^s_1 = \begin{cases} q^s_t, & \text{if } s \text{ matched} \\ q^s_2, & \text{if no match: } q^s_t = \tau \\ \hat{q}, & \text{if new track} \end{cases},$$

being the matched detection,

$$\xi'^s_1 = \begin{cases} 1, & \text{if } s \text{ matched} \\ 0, & \text{if no match: } q^s_t = \tau \\ 1, & \text{if new track} \end{cases},$$

being the updated valid flag, and

$$\psi'^s = \begin{cases} \psi_{max} + 1, & \text{if new track} \\ \psi^s, & \text{otherwise} \end{cases},$$

the updated ID.

3.7. Training

We train our network in two phases. First, we train the original SDD on a detection dataset. In the second phase, we add the RNN and train on sequences of 16 frame minimizing the additional tracking losses. The tracking objective consist of two goals: find each object in the scene and keep the ID for each track constant. Therefore we use a combination of multiple loss functions consisting of the loss for the affinity metric $\mathcal{L}_m$, the loss for the new detection score $\mathcal{L}_c$, the loss for the association metric $\mathcal{L}_o$, and the loss for the track score $\mathcal{L}_l$. For the affinity metric and association metric we use the sigmoid cross entropy loss, independently normalized for positive and negative samples:

$$\mathcal{L}_x = -\frac{1}{|N_x^+|} \sum_{n \in N_x^+} p_n^x \log \hat{p}_n^x - \frac{1}{|N_x^-|} \sum_{n \in N_x^-} (1 - p_n^x) \log (1 - \hat{p}_n^x),$$

with N_x^+ the set of matching sample pairs, N_x^- the set of negative sample pairs, and p_n^x ground truth. For the affinity metric the estimation is the sigmoid of the affinity measure: $\hat{p}^{\text{m}}_{s,d,\Delta} = \sigma(m_{s,d,\Delta})$, and likewise for the association metric: $\hat{p}^{\text{o}}_{s,d^*} = \sigma(o_{s,d^*})$. Input sample triplets ((s, d, Δ) for $\mathcal{L}_m$), sample pairs ((s, d^*) for $\mathcal{L}_o$) and their respective ground truth ($p^{\text{m}}_{s,d,\Delta}$ and p^{o}_{s,d^*}), are generated from a selected random set of ground truth tracks $\mathcal{S}^{gt}$. In every frame we sample all triplets (s, d, Δ) with $s \in [1, |\mathcal{S}^{gt}|]$, $d \in [1, |\mathcal{D}|]$, and $\Delta \in [1, \Delta_{max}]$. A sample triplet is positive, $(s, d, \Delta) \in N^+_{\text{m}}$ and $p^{\text{m}}_{s,d,\Delta} = 1$, if the overlap between the detection box $q^d_{\text{box}} = (x^d_t, y^d_t, w^d_t, h^d_t)$ with the ground truth track bounding box $q^s_{\text{box}} = (x^s_t, y^s_t, w^s_t, h^s_t)$ is sufficiently high, *i.e.*:

$$\frac{q^d_{\text{box}} \cap q^s_{\text{box}}}{q^d_{\text{box}} \cup q^s_{\text{box}}} > 0.85,$$

and if the track was visible in frame $t - \Delta$, with t being the current frame. If the overlap is lower than 0.5 the sample is regarded as negative, *i.e.* $(s, d, \Delta) \in N^-_{\text{m}}$ and $p^{\text{m}}_{s,d,\Delta} = 0$, given that the track s was visible in frame $t - \Delta$.

The ground truth for the association metric is collected in similar way. We sample track detection pairs (s, d^*) from the current frame, with $s \in [1, |\mathcal{S}^{gt}|]$, $d^* \in [1, |\mathcal{D}^*|]$. $\mathcal{D}^*$ is the set of filtered detections generated by the NMS. Again a pair is regarded as positive if the overlap between the track ground truth bounding box and the detection bounding box is high and negative when the overlap is low.

For the loss of the re-weighted detection score $\mathcal{L}_c$ and the loss for the track score $\mathcal{L}_l$ we use the sigmoid cross entropy loss

$$\mathcal{L}_x = \frac{-1}{|N_x|} \sum_{n \in N_x} \left[p_n^x \log \hat{p}_n^x + (1 - p_n^x) \log (1 - \hat{p}_n^x) \right],$$

where N_c is a set of detections $N_c \subset \mathcal{D}$, $\hat{p}^c_d = \sigma(c'^d)$ is the sigmoid of the re-weighted detection score with their respective ground truth p^c_d. $N_l = \mathcal{S}^{gt}$ is the set of sampled ground truth tracks, $\hat{p}^l_s = \sigma(l^s)$ is the sigmoid of the track score with their ground truth p^l_s. Similar to the other losses the ground truth is positive, $p^c_d = 1$, if the detection has a high overlap with any annotated track bounding box and negative, $p^c_d = 0$, if the overlap is low. Analog for p^l_s with the restriction that $p^l_s = 1$ only if the ID of the track is correct.

For a faster convergence, we correct wrong associations before continuing with the next frame and tracks are only initialized from the ground truth set.

4. Evaluation

We test our method on the DETRAC[26] and on the MOT16[15] benchmarks. The DETRAC benchmark includes 10 hours of videos in traffic surveillance with annotated vehicle tracks. From the DETRAC training set we

	MOTA	IDs	Rcll	Prcn	MT	ML	FM	MOTP
$\Delta_{max} = 6$	69.3	443	76.0	92.6	61.0%	4.8%	675	82.8
$\Delta_{max} = 4$	**72.5**	346	**79.2**	92.7	**68.1%**	**3.6%**	547	81.7
$\Delta_{max} = 2$	68.1	311	73.1	**94.1**	55.2%	5.7%	781	**83.5**
$\Delta_{max} = 1$	70.0	474	75.9	93.5	59.2%	4.8%	686	83.1
$\Delta_{max} = 1$, no app	71.7	116	77.5	93.2	62.5%	4.2%	783	82.7
CEM[1]	15.7	**65**	25.9	72.0	4.8%	57.8%	**100**	74.8
GOG[17]	49.1	328	70.1	77.3	44.9%	9.6%	472	66.0
IHTLS[8]	62.6	205	78.3	83.5	63.4%	3.8%	563	82.2
RMOT[28]	62.6	132	69.4	91.4	42.1%	6.5%	174	75.7

Table 1: Results on our DETRAC validation set. Best result indicated in bold.

use 52 sequences for training and eight for testing[1]. We first train the SSD for detecting vehicles. Some sequences contain regions that are ignored in the benchmark. We included these regions into the training so that bounding box results that overlap with these regions do not influence the training losses. We set the input size of the detector to 480x270 pixel and train for 140000 iterations with a batch size of 16 and a learn rate of 0.001, using SGD for optimization. In the second phase we add appearance features ($\mathcal{A}$, with $|\mathcal{A}| = 32$) and the association RNN and train on sequences of 16 frames for 35000 iterations. Because of memory limitations, we select a set of eight ground truth tracks for each sequence.

For testing we set the maximum number of tracks $|\mathcal{S}|$ to 35 and evaluated different sizes of track history Δ_{max}. Further we test the method by omitting the appearance features with $\Delta_{max} = 1$, so that only the distance to the last track position is used. We compare our method to the benchmark provided implementation of other tracking by detection methods. As detector for the other methods, we use our trained SSD after the first phase without the RNN but with the original non-maximum suppression included. Since the other methods are depended on the detection score, we evaluated each method 10 times with a different threshold to filter low scoring detections and report the best result. For our method, we filter tracking bounding boxes with a tracking score lower than zero. We use the evaluation tool provided from the DETRAC benchmark[26] and report the following metrics: multi-object tracking accuracy (MOTA)[4], the number of ID switches (IDs), recall (Rcll) and precision (Prcn) of the tracking bounding boxes, percentage of mostly tracked (MT) and mostly lost (ML) of the tracks, the number of fragments (FM) and the multi-object tracking precision (MOTP)[4]. Results are summarized in table 1.

The results show that the MOTA of our method is stable most of the time, but larger than the MOTA of the compared methods. Further, the number of ID switches is significant lower when not using appearance features. We attribute this to the fact that the association metric $m_{d,s,\Delta}$ is more difficult to train for positive pairs when using the appearance features. In future, we plan to replace the fixed threshold for the association by a learned parameter. Overall, the number of fragments is smaller for our method when using appearance features, even with varied recall. This shows that the appearance features help to associate detection when the position difference becomes too high. For our $\Delta_{max} = 4$ setting the runtime of the whole method for each frame is 62ms (16.2 frames per second). Using only the SSD network part the runtime is 37.2ms (26.9 frames per second).

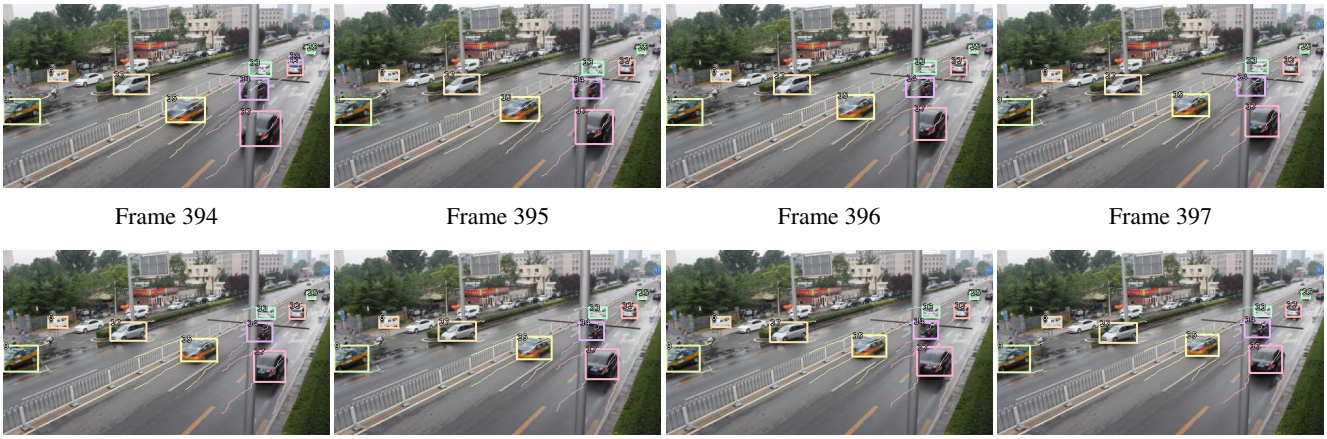
The MOT16[15] benchmark consists of pedestrian tracking sequences. They differ in image resolution and camera movement. In the first phase we train the SSD to detect pedestrians on the training sets of MOT15[12], MOT16[15], Caltech Pedestrian[9] and CityPersons[30]. We set the detector input size to 480x270 pixel and train for 250000 iterations with a batch size of 16 and a learn rate of 0.001, using SGD for optimization. In the second phase we train on the MOT15 and MOT16 training set on sequences of 16 frames for 45000 iterations with eight ground truth tracks for each sequence. For testing we set the maximum number of tracks $|\mathcal{S}|$ to 55 and the size of the track history to $\Delta_{max} = 4$. We filter tracking bounding boxes if the tracking score is lower than zero.

We compare our method with published state of the art online tracking methods on the MOT16 test set (see table 2). For a fair comparison between methods, the annotations for the test set are not public. Instead, each author submits the tracking results to the benchmark website to get results. Different to the DETRAC benchmark, we report the absolute number of false positive (FP) and false negative (FN) instead of precision and recall. The results show, that our method does not reach top performance but can best several other methods. Because of the increased number of tracks, our runtime increases to 222ms per frame (4.5 frames per second) for the complete detection and tracking method.

[1]Validation set sequence numbers: 39781, 40152, 40181, 40752, 41063, 41073, 63521, and 63525

	MOTA	IDs	FP	FN	MT	ML	Runtime
POI[29]	**66.1**	805	5,061	**55,914**	**34.0%**	**20.8%**	9.9 Hz
SORTwHPD16[5]	59.8	1,423	8,698	63,245	25.4%	22.7%	**59.5 Hz**
EAMTT[19]	52.5	910	4,407	81,223	19.0%	34.9%	12.2 Hz
AMIR[18]	47.2	774	**2,681**	92,856	14.0%	41.6%	1.0 Hz
STAM16[7]	46.0	473	6,895	91,117	14.6%	43.6%	0.2 Hz
CDA_DDALv2[2]	43.9	676	6,450	95,175	10.7%	44.4%	0.5 Hz
oICF[11]	43.2	**381**	6,651	96,515	11.3%	48.5%	0.4 Hz
$\Delta_{max} = 4$	39.1	1,906	9,411	99,727	11.1%	41.1%	4.5* Hz
EAMTT_pub[19]	38.8	965	8,114	102,452	7.9%	49.1%	11.8 Hz
JCmin_MOT[6]	36.7	667	2,936	111,890	7.5%	54.4%	14.8 Hz
HISP_T[3]	35.9	2,594	6,412	107,918	7.8%	50.1%	4.8 Hz
GMPHD_HDA[22]	30.5	539	5,169	120,970	4.6%	59.7%	13.6 Hz

Table 2: Results on MOT16 test set from published online methods. Best result indicated in bold. *Contrary to other methods, our runtime includes the detection phase.



Frame 394 Frame 395 Frame 396 Frame 397

Frame 398 Frame 399 Frame 400 Frame 401

Figure 3: Tracking through partial occlusion on DETRAC sequence 40152.

5. Conclusion

We presented a joint detection and online multi-object tracking method, which expands the SSD with appearance features and a recurrent network for track-detection association. We showed initial results on the DETRAC and MOT16 benchmark. On the DETRAC benchmark we archive competitive results. On the MOT16 benchmark, our method reports a high number of false positives and ID switches. Partly both numbers can be explained by the high sensitivity of our detection method, but also by the fix threshold for the association metric $m_{d,s,\Delta}$ yielding too few associations. Further currently the method has no technique to predict movements of a track during occlusion or to re-identify an object when it is occluded for more than Δ_{max} frames. Nevertheless, we believe that the overall framework shows potential in the cooperation between detection and tracking method.

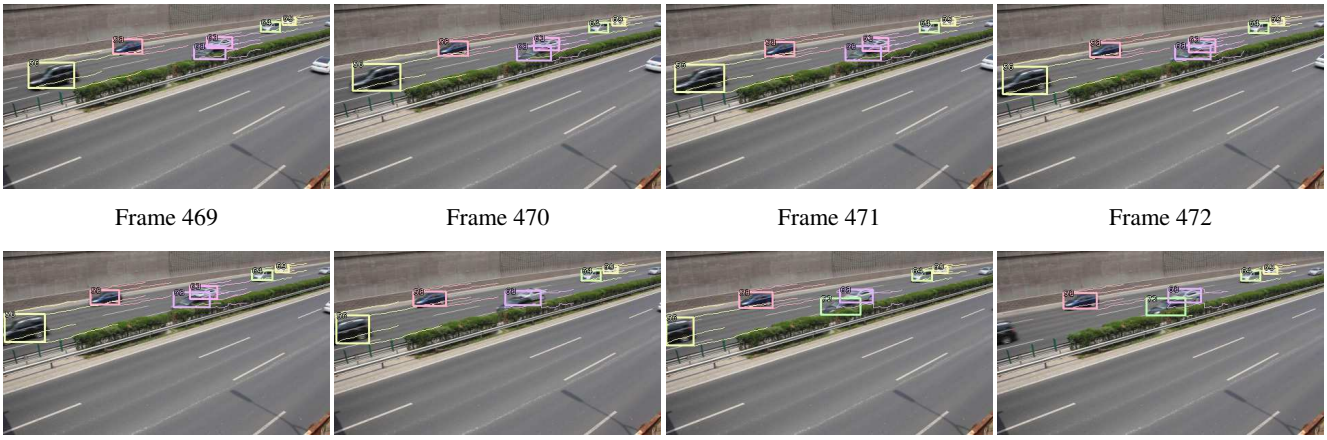
In future work we want to expand the idea of a fully integrated multi-object tracking network. For that, we plan to further decrease the impact of the non-maximum suppression and to include a method to re-identify objects after occlusions.

References

[1] A. Andriyenko and K. Schindler. Multi-target tracking by continuous energy minimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1265–1272, 2011. 5

[2] 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*, 40:595–610, 2018. 2, 6

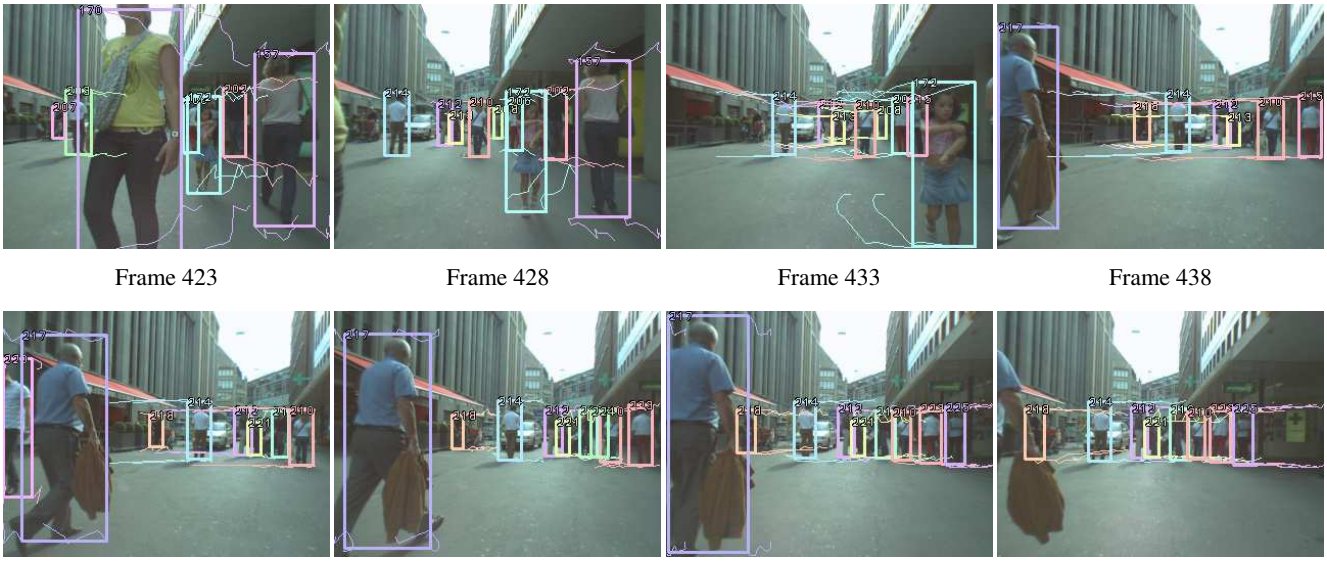
[3] N. L. Baisa et al. Online multi-target visual tracking using a hisp filter. In *Proceedings of the 13th International*



Frame 469 Frame 470 Frame 471 Frame 472

Frame 473 Frame 474 Frame 475 Frame 476

Figure 4: Typical ID switch error on DETRAC sequence 41073. A misaligned detection in frame 474 causes ID switch.



Frame 423 Frame 428 Frame 433 Frame 438

Frame 443 Frame 448 Frame 453 Frame 458

Figure 5: Tracking through high camera movement on sequence MOT16_06. Typical ID switch error after occlusion.

Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP), 2018. 6

[4] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *Journal on Image and Video Processing*, 2008. 5

[5] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016. 2, 6

[6] A. Boragule and M. Jeon. Joint cost minimization for multi-object tracking. In *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2017. 6

[7] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2, 6

[8] C. Dicle, O. I. Camps, and M. Sznaier. The way they move: Tracking multiple targets with similar appearance. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2304–2311, 2013. 5

[9] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:743–761, 2012. 5

[10] C. Feichtenhofer, A. Pinz, and A. Zisserman. Detect to track and track to detect. In *Proceedings of the IEEE Conference*

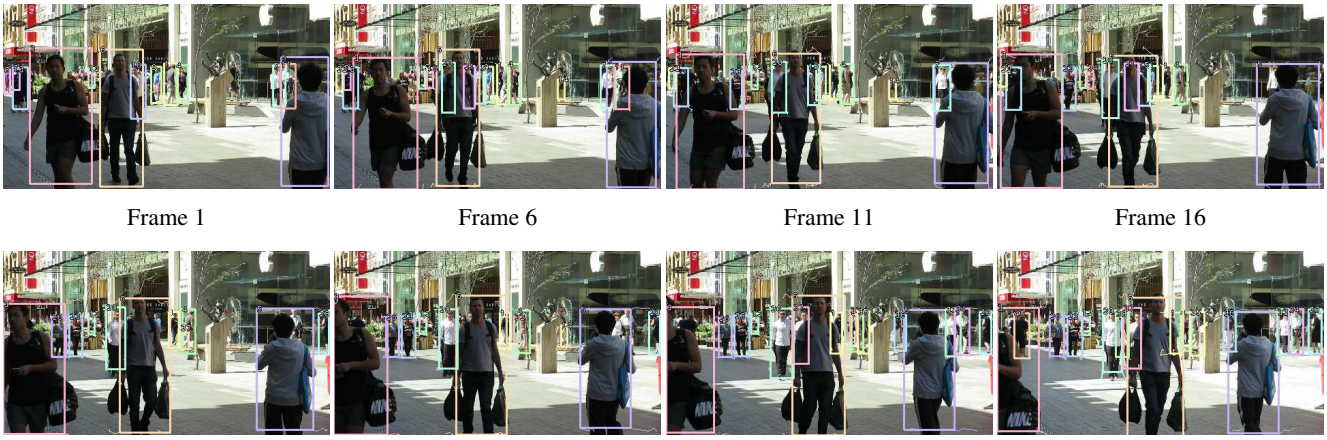


Figure 6: High number of false positive bounding boxes on sequence MOT16_08, because of overlapping pedestrian and background clutter.

on Computer Vision and Pattern Recognition (CVPR), pages 3038–3046, 2017. 2

[11] H. Kieritz, S. Becker, W. Hübner, and M. Arens. Online multi-person tracking using integral channel features. In *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 122–130, 2016. 1, 6

[12] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. MOTChallenge 2015: Towards a benchmark for multi-target tracking. *arXiv*, 2015. 5

[13] L. Leal-Taixé, A. Milan, K. Schindler, D. Cremers, I. Reid, and S. Roth. Tracking the Trackers: An Analysis of the State of the Art in Multiple Object Tracking. *arXiv*, 2017. 1

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 21–37, 2016. 1, 2

[15] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv*, 2016. 1, 4, 5

[16] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler. Online multi-target tracking using recurrent neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 2

[17] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1201–1208, 2011. 5

[18] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 2, 6

[19] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro. Multi-target tracking with strong and weak detections. In *Proceedings of the ECCV Workshops-Benchmarking Multi-Target Tracking*, volume 5, 2016. 6

[20] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro. Online multi-target tracking with strong and weak detections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 84–99, 2016. 1

[21] J. Son, M. Baek, M. Cho, and B. Han. Multi-object tracking with quadruplet convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5620–5629, 2017. 2

[22] Y.-m. Song and M. Jeon. Online multiple object tracking with the hierarchically adopted gm-phd filter using motion and appearance. In *Proceedings of the IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, 2016. 6

[23] P. Tang, C. Wang, X. Wang, W. Liu, W. Zeng, and J. Wang. Object detection in videos by short and long range object linking. *arXiv*, 2018. 2

[24] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph decomposition for multi-target tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5033–5041, 2015. 2

[25] S. Tang, M. Andriluka, B. Andres, and B. Schiele. Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3539–3548, 2017. 2

[26] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M. Yang, and S. Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *arXiv*, 2015. 1, 4, 5

[27] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4705–4713, 2015. 1

[28] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon. Bayesian multi-object tracking using motion context from multiple objects. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 33–40, 2015. 5

[29] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan. Poi: multiple object tracking with high performance detection and appearance feature. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 36–42, 2016. 2, 6

[30] S. Zhang, R. Benenson, and B. Schiele. Citypersons: A diverse dataset for pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3221, 2017. 5