

Object Tracking

Tomasz Stańczyk
tomasz.stanczyk@inria.fr



With inspiration and some images from the lecture of Prof. Dr. Laura Leal-Taixé:
<https://www.youtube.com/watch?v=mrMspzKcOAM>

3iA Côte d'Azur
Interdisciplinary Institute
for Artificial Intelligence

Two types of tracking discussed today

Visual-object tracking - briefly

Multi-object tracking - major part of the lecture

Visual-object tracking (single-object tracking)

Given a video, find out which parts of the image depict the same object in different frames

To give you an idea: <https://www.youtube.com/watch?v=lqMgsiU9B5E>

It can be applied in surveillance (tracking the target of interest), observation applications (e.g. animals) and so on

Correlation tracker

Implemented in dlib library: <http://dlib.net/>

Easy to install and run

Previous video - that was the correlation tracker!

Curious? Try it yourself:

http://dlib.net/correlation_tracker.py.html

All instructions provided in the file

Based on the paper: "Accurate Scale Estimation for Robust Visual Tracking"

<http://www.bmva.org/bmvc/2014/files/paper038.pdf>

Multi-object tracking

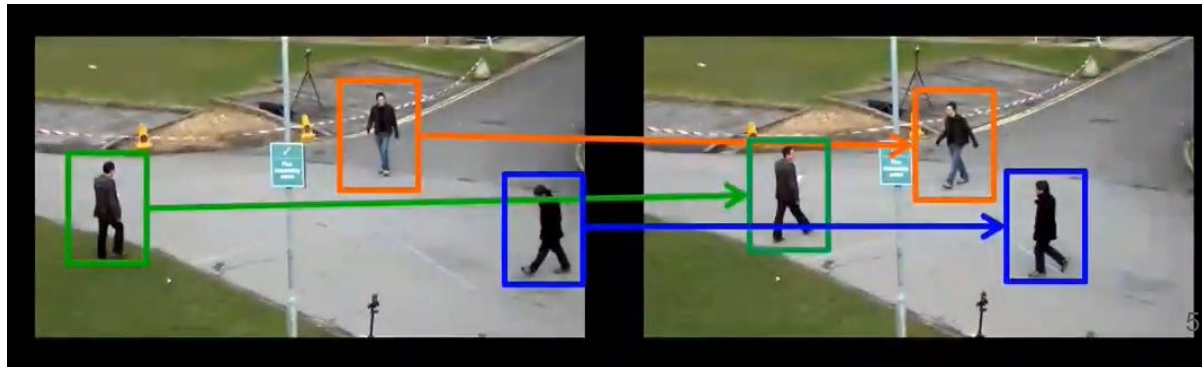
Detect and track all objects in a scene

(Again) Given a video, find out which parts of the image depict the same object in different frames

Detectors are often used as starting points -> Tracking by detection

Creating tracklets

Assigning ID to the objects



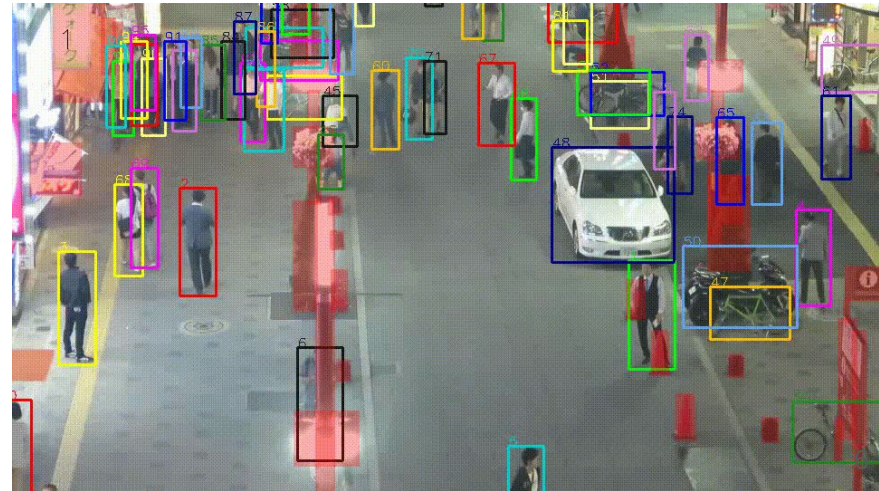
The importance of tracking

Pointing the objects when detection fails

- occlusions
- variations in viewpoint, pose, blur and illumination between frames of a sequence
- background clutter

Keeping the track of the object(s) of interest
(detections can be returned with a random order per each frame, without any ID information)

Reasoning about the dynamic world,
e.g. trajectory prediction



What tracking is about

Similarity measurement

Correlation

Correspondence

Matching/retrieval

Data association

What tracking is about

Learning to model the target:

- appearance - how the target looks like
 - * single-object tracking
 - * re-identification
- motion - predicting where the target goes
 - * trajectory prediction

Challenges

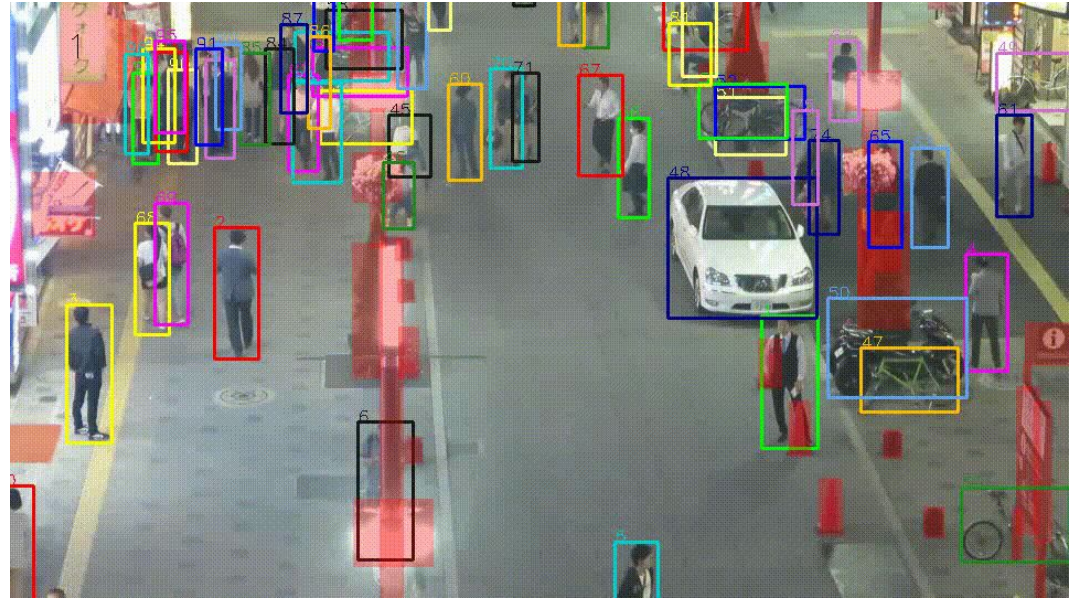
Multiple object of the same type

Heavy occlusions

Often very similar appearance

Emerging issues

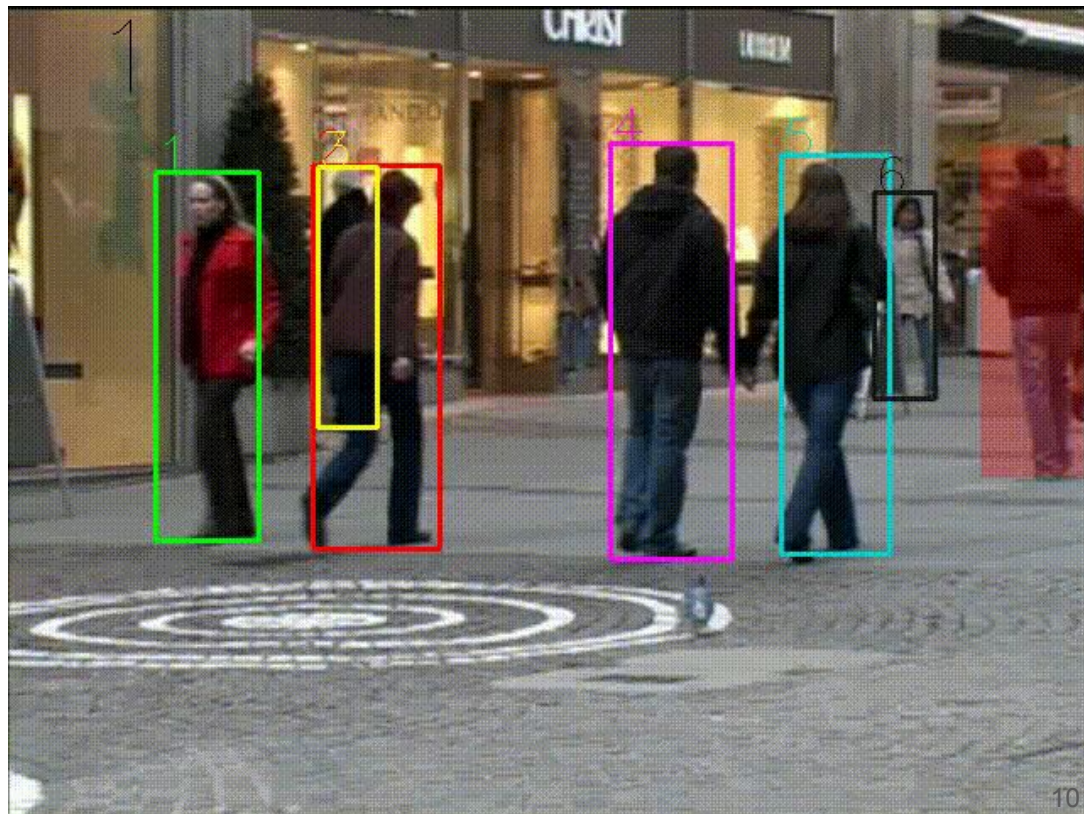
- identity switches,
- short tracklets, fragmented tracklets
- targets leaving the scene (and then coming back)



MOT Challenge - MOT15

<https://motchallenge.net/>

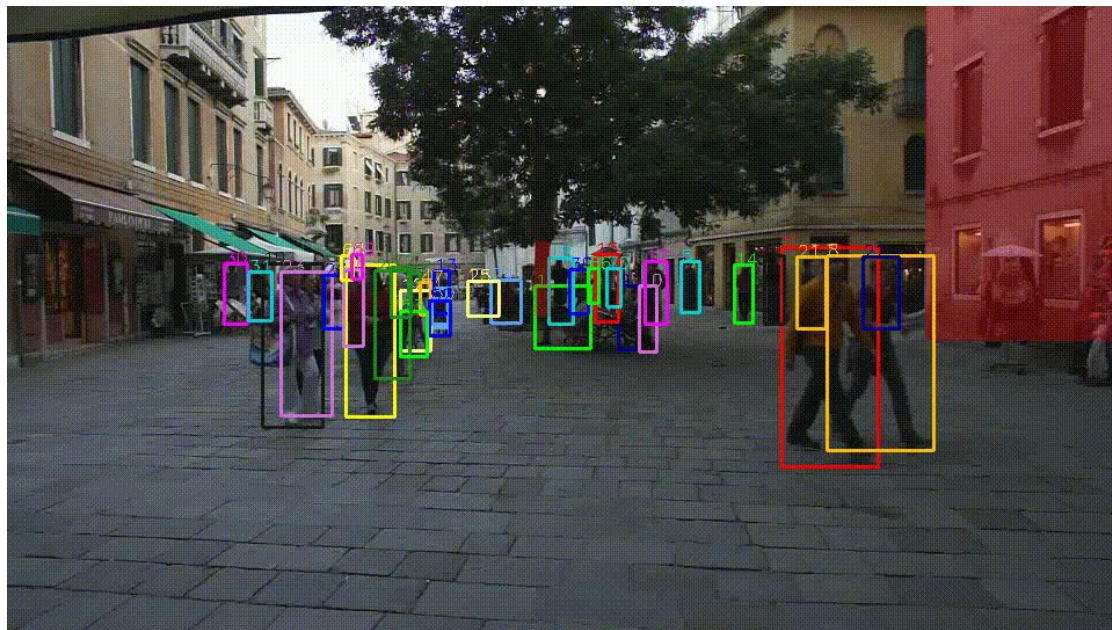
Multi-object tracking challenge



MOT Challenge - MOT17

<https://motchallenge.net/>

Multi-object tracking challenge



MOT Challenge - MOT20

<https://motchallenge.net/>

Multi-object tracking challenge



Online tracking and offline tracking

Online tracking: processing frames as they become available

- real-time application, e.g. autonomous driving, AR/VR
- prone to drifting - hard to recover from errors or occlusions

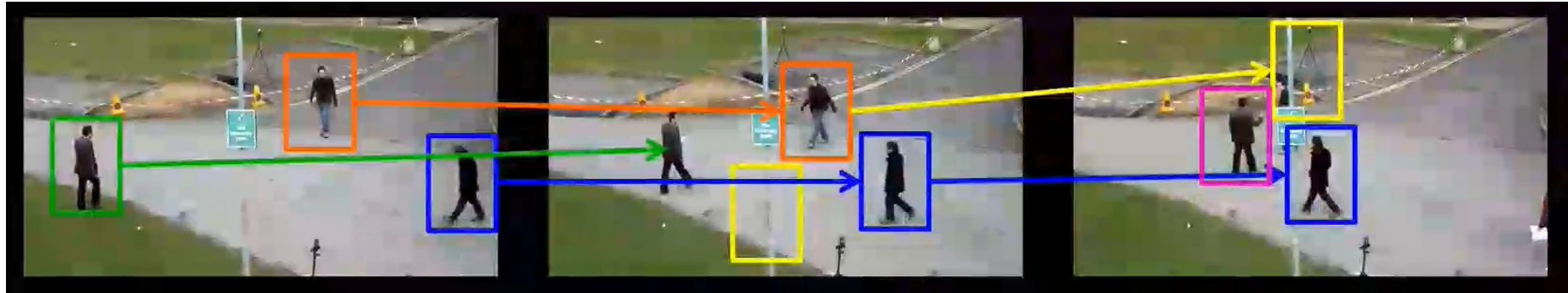
Offline tracking: processing a batch of frames

- good to recover from (short) occlusions
- non suitable for real-time applications
- yet suitable for video analysis, automatic labeling, video editing

Paradigm: Tracking by detection

Detection - detector is run on each frame to obtain a set of proposed locations

Data association - connecting the detections in the temporal domain to create trajectories



A simple online tracker

Track initialization, e.g. using a detector

Prediction of the next position - motion model

Matching predictions with detections - appearance model

A simple online tracker

Prediction of the next position - motion model









- Kalman filter
- Recurrent architectures
- simple constant velocity model

Bipartite matching

Define distance between boxes, e.g. IoU, pixel distance, reID

Solve the unique matching, e.g. with the Hungarian algorithm

Solutions are the unique assignments that minimize the total cost

				
	0.9	0.8	0.8	0.1
	0.5	0.4	0.3	0.8
	0.2	0.1	0.4	0.8
	0.1	0.2	0.5	0.9

The role of learning

Track initialization, e.g. using a detector

- Deep learning based detectors

Prediction of the next position - motion model

- Adding temporal complexity

Matching predictions with detections

- adding feature complexity
- improving appearance models - re-identification

Adding computational complexity

Tracking by detection - DeepSORT

"Simple Online and Realtime Tracking with a Deep Association Metric"

<https://arxiv.org/pdf/1703.07402.pdf>

Simple approach and fast

Important milestone for MOT development

(Used to be) widely used, e.g. by companies

Not suitable for challenging scenarios

Many identity switches

Listing 1 Matching Cascade

Input: Track indices $\mathcal{T} = \{1, \dots, N\}$, Detection indices $\mathcal{D} = \{1, \dots, M\}$, Maximum age A_{\max}

- 1: Compute cost matrix $\mathbf{C} = [c_{i,j}]$ using Eq. 5
- 2: Compute gate matrix $\mathbf{B} = [b_{i,j}]$ using Eq. 6
- 3: Initialize set of matches $\mathcal{M} \leftarrow \emptyset$
- 4: Initialize set of unmatched detections $\mathcal{U} \leftarrow \mathcal{D}$
- 5: **for** $n \in \{1, \dots, A_{\max}\}$ **do**
- 6: Select tracks by age $\mathcal{T}_n \leftarrow \{i \in \mathcal{T} \mid a_i = n\}$
- 7: $[x_{i,j}] \leftarrow \text{min_cost_matching}(\mathbf{C}, \mathcal{T}_n, \mathcal{U})$
- 8: $\mathcal{M} \leftarrow \mathcal{M} \cup \{(i, j) \mid b_{i,j} \cdot x_{i,j} > 0\}$
- 9: $\mathcal{U} \leftarrow \mathcal{U} \setminus \{j \mid \sum_i b_{i,j} \cdot x_{i,j} > 0\}$
- 10: **end for**
- 11: **return** \mathcal{M}, \mathcal{U}

Tracking by detection - ByteTrack

"ByteTrack: Multi-Object Tracking by Associating Every Detection Box"

<https://arxiv.org/pdf/2110.06864.pdf>

Algorithm 1: Pseudo-code of BYTE.

```
Input: A video sequence  $V$ ; object detector  $Det$ ; detection score threshold  $\tau$   
Output: Tracks  $\mathcal{T}$  of the video  
1 Initialization:  $\mathcal{T} \leftarrow \emptyset$   
2 for frame  $f_k$  in  $V$  do  
   /* Figure 2(a) */  
   /* predict detection boxes & scores */  
   3  $\mathcal{D}_k \leftarrow Det(f_k)$   
   4  $\mathcal{D}_{high} \leftarrow \emptyset$   
   5  $\mathcal{D}_{low} \leftarrow \emptyset$   
   6 for  $d$  in  $\mathcal{D}_k$  do  
     7 if  $d.score > \tau$  then  
       8 |  $\mathcal{D}_{high} \leftarrow \mathcal{D}_{high} \cup \{d\}$   
     9 end  
    10 else  
      11 |  $\mathcal{D}_{low} \leftarrow \mathcal{D}_{low} \cup \{d\}$   
    12 end  
  13 end  
  
   /* predict new locations of tracks */  
  14 for  $t$  in  $\mathcal{T}$  do  
    15 |  $t \leftarrow KalmanFilter(t)$   
  16 end  
  
   /* Figure 2(b) */  
   /* first association */  
  17 Associate  $\mathcal{T}$  and  $\mathcal{D}_{high}$  using Similarity#1  
  18  $\mathcal{D}_{remain} \leftarrow$  remaining object boxes from  $\mathcal{D}_{high}$   
  19  $\mathcal{T}_{remain} \leftarrow$  remaining tracks from  $\mathcal{T}$   
  
   /* Figure 2(c) */  
   /* second association */  
  20 Associate  $\mathcal{T}_{remain}$  and  $\mathcal{D}_{low}$  using similarity#2  
  21  $\mathcal{T}_{re-remain} \leftarrow$  remaining tracks from  $\mathcal{T}_{remain}$   
  
   /* delete unmatched tracks */  
  22  $\mathcal{T} \leftarrow \mathcal{T} \setminus \mathcal{T}_{re-remain}$   
  
   /* initialize new tracks */  
  23 for  $d$  in  $\mathcal{D}_{remain}$  do  
    24 |  $\mathcal{T} \leftarrow \mathcal{T} \cup \{d\}$   
  25 end  
  26 end  
  27 Return:  $\mathcal{T}$ 
```

Tracking by detection - ByteTrack

Processing on the fly

Fine-tuned YOLOX object detector

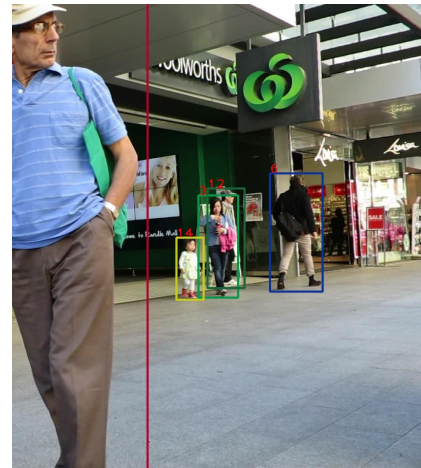
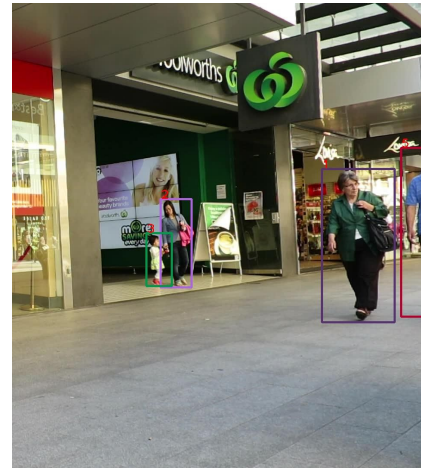
IoU, Kalman filter, well-engineered algorithm

No learning scheme

Very good baseline

Many identity switches

Fragmented tracklets



Tracking by detection - SUSHI

"Unifying Short and Long-Term Tracking with Graph Hierarchies"

<https://arxiv.org/pdf/2212.03038.pdf>

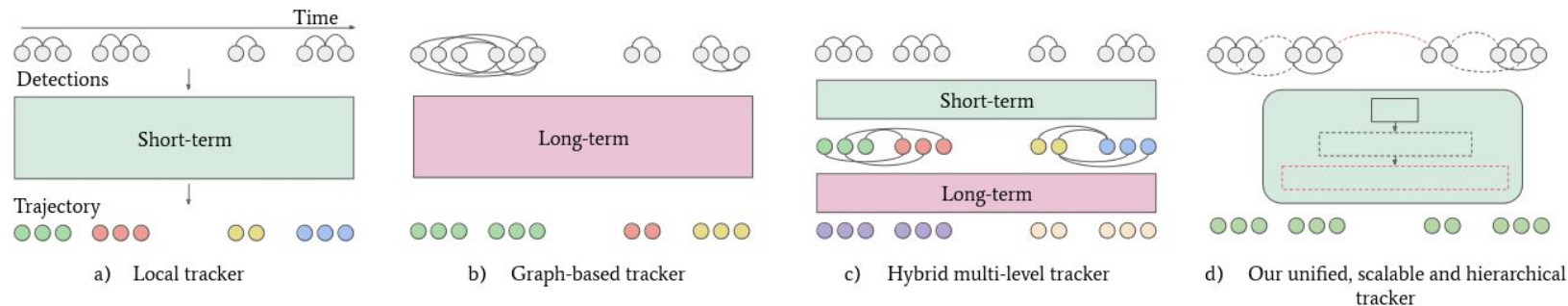


Figure 1. (a) Local tracker focusing on short-term scenarios and lacking robustness at long-term identity preservation (b) Graph-based tracker tackling longer-term association but unable to cover large time gaps due to its limited scalability (c) Hybrid multi-level tracker engineering a combination of techniques but still struggling with scalability (d) Our unified hierarchical tracker with high scalability.

Tracking by detection - SUSHI

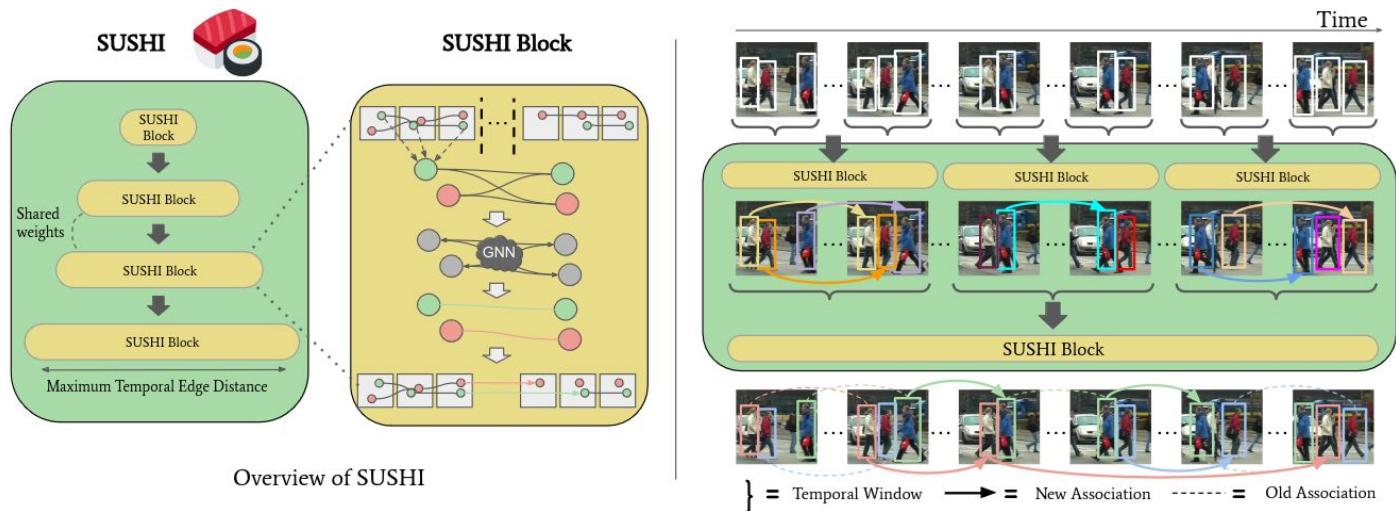


Figure 2. SUSHI consists of a set of SUSHI blocks operating hierarchically over a set of tracklets (with initial length one) in a video clip. Each SUSHI block considers a graph with tracklets from a subclip as nodes, performs neural message passing over it, and merges nodes into longer tracks. Over several hierarchy levels SUSHI blocks are able to progressively merge tracklets into tracks spanning over the entire clip. Notably, *SUSHI blocks share the same GNN architecture and weights*, hence making SUSHI unified across temporal scales.

Tracking by detection - SUSHI

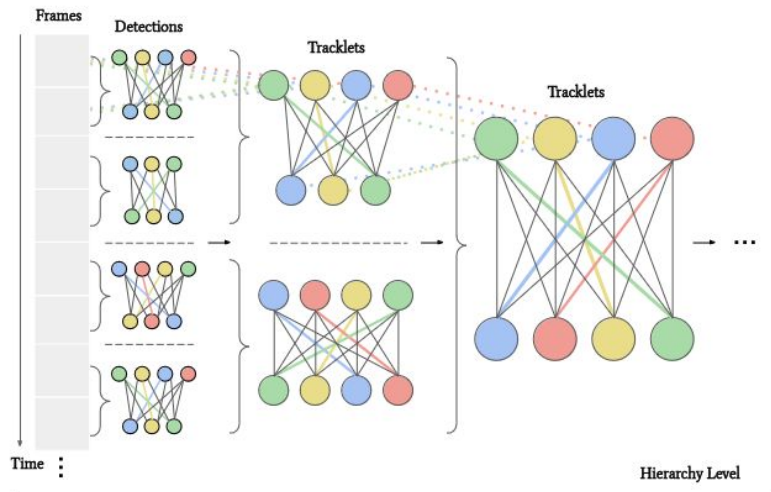


Figure 3. Our hierarchy is based on recursive partitioning of the video clip and we only allow edges within these partitions. After each level, we merge tracklets belonging to the same identity and consider edges spanning across longer timespans.

Tracking by detection - SUSHI

Global optimization approach (it needs to see all the frames in advance)

Graph-based association method

Also built on YOLOX (as ByteTrack)

Feature extraction through re-iD and mathematical derivations

Encoding the features further in the architecture

Behaving exceptionally well

Handling many challenging cases, e.g. occlusion, crowded groups, longer trajectories

Some identity switches still present

Tendency to link newly appearing people to those who have left the scene

Still some Fragmented tracklets

Method	Det Ref.	IDF1 \uparrow	HOTA \uparrow	MOTA \uparrow	ID Sw. \downarrow
MOT17 - Public					
Tracktor [3]	Tracktor	55.1	44.8	56.3	1987
LPT [26]	Tracktor	57.7	–	57.3	1424
MPNTrack [6]	Tracktor	61.7	49.0	58.8	1185
Lif_T [17]	Tracktor	65.6	51.3	60.5	1189
ApLift [18]	Tracktor	65.6	51.1	60.5	1709
GMT [16]	Tracktor	65.9	51.2	60.2	1675
LPC_MOT [10]	Tracktor	66.8	51.5	59.0	1122
SUSHI (Ours)	Tracktor	71.5	54.6	62.0	1041
MOT17 - Private					
QDTrack [33]	\times	66.3	53.9	68.7	3378
TrackFormer [30]	\times	68.0	57.3	74.1	2829
MOTR [65]	\times	68.6	57.8	73.4	2439
PermaTrack [47]	\times	68.9	55.5	73.8	3699
MeMOT [8]	\times	69.0	56.9	72.5	2724
GTR [70]	\times	71.5	59.1	75.3	2859
FairMOT [68]	\times	72.3	59.3	73.7	3303
GRTU [49]	\times	75.0	62.0	74.9	1812
CorrTracker [48]	\times	73.6	60.7	76.5	3369
Unicorn [60]	\times	75.5	61.7	77.2	5379
ByteTrack [†] [67]	\times	77.1	62.8	78.9	2363
ByteTrack [67]	\times	77.3	63.1	80.3	2196
SUSHI (Ours)	\times	83.1	66.5	81.1	1149

Table 2. Test set results on MOT17 benchmark. Det. Ref. denotes the public detection refinement strategy. As ByteTrack (gray) uses different thresholds for test set sequences and interpolation, we also report scores by disabling these as ByteTrack[†] (black).

Tracking by detection concluded

Leverages well the advances in object detection

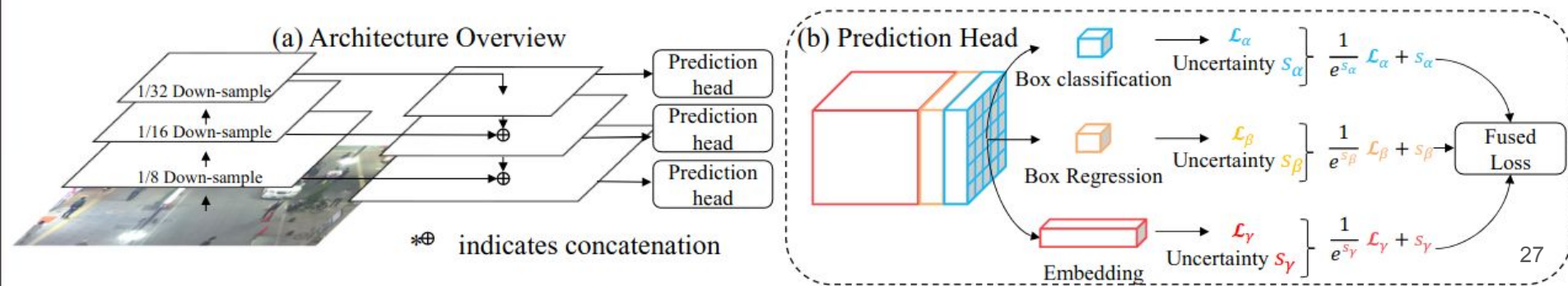
It can be used online (Hungarian) + by batches (adding computational complexity)

Paradigm: Joint detection and tracking

Joint detection and association embedding (JDE) - anchor based

"Towards Real-Time Multi-Object Tracking"

<https://arxiv.org/pdf/1909.12605.pdf>

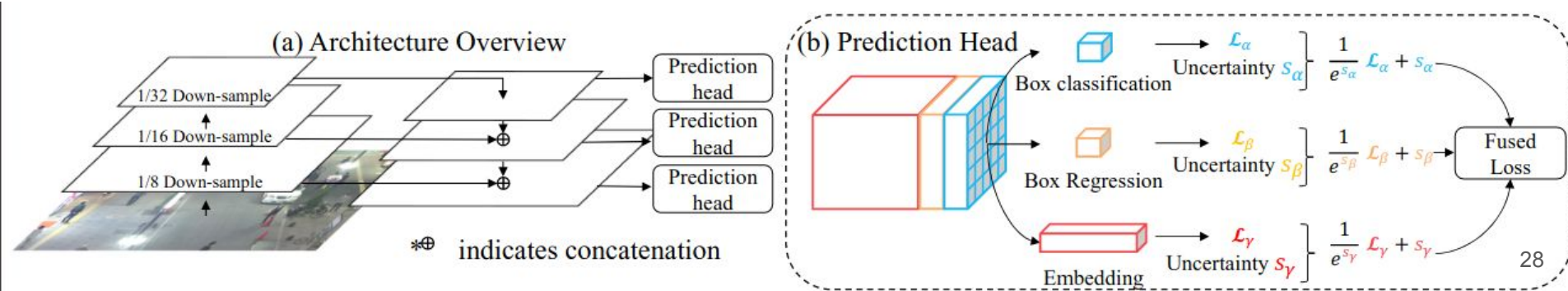


Joint detection and association embedding (JDE) - anchor based

Association via embedding distance

Near-real time (shared backbone)

Jointly training for detection and tracking, but tasks still separated in different heads



Anchor-free JDE - FairMOT

"FairMOT: On the Fairness of Detection and Re-Identification in Multiple Object Tracking", <https://arxiv.org/pdf/2004.01888.pdf>

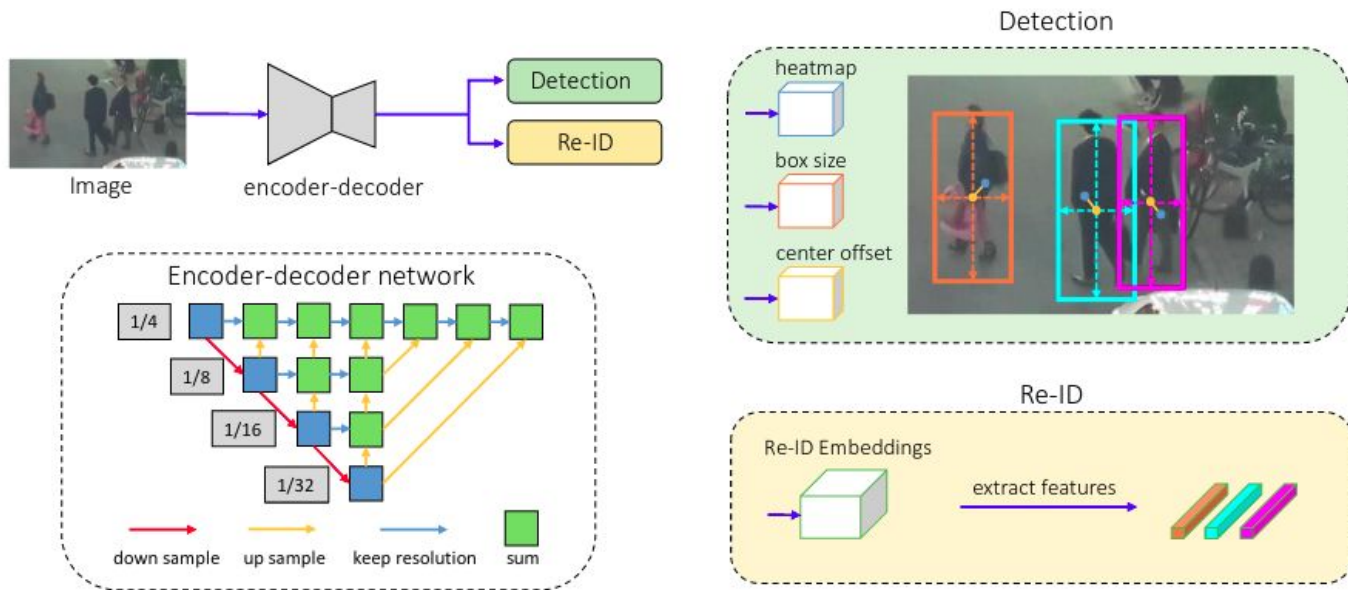
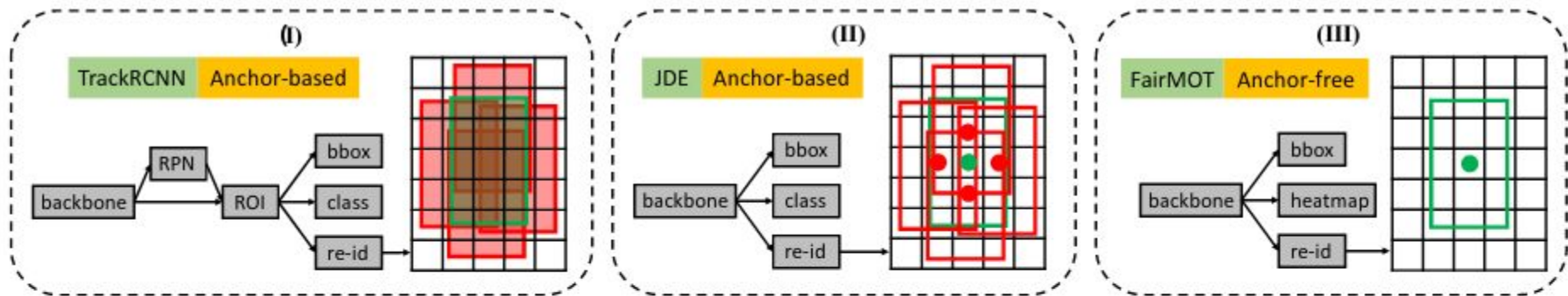


Fig. 1 Overview of our one-shot tracker *FairMOT*. The input image is first fed to an encoder-decoder network to extract high resolution feature maps (stride=4). Then we add two homogeneous branches for detecting objects and extracting re-ID features, respectively. The features at the predicted object centers are used for tracking.

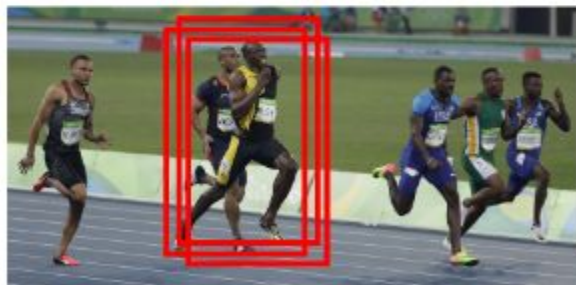
Anchor-free JDE - FairMOT



(a) Comparison of the existing one-shot trackers and FairMOT



(b) One anchor contains multiple identities



(c) Multiple anchors response for one identity

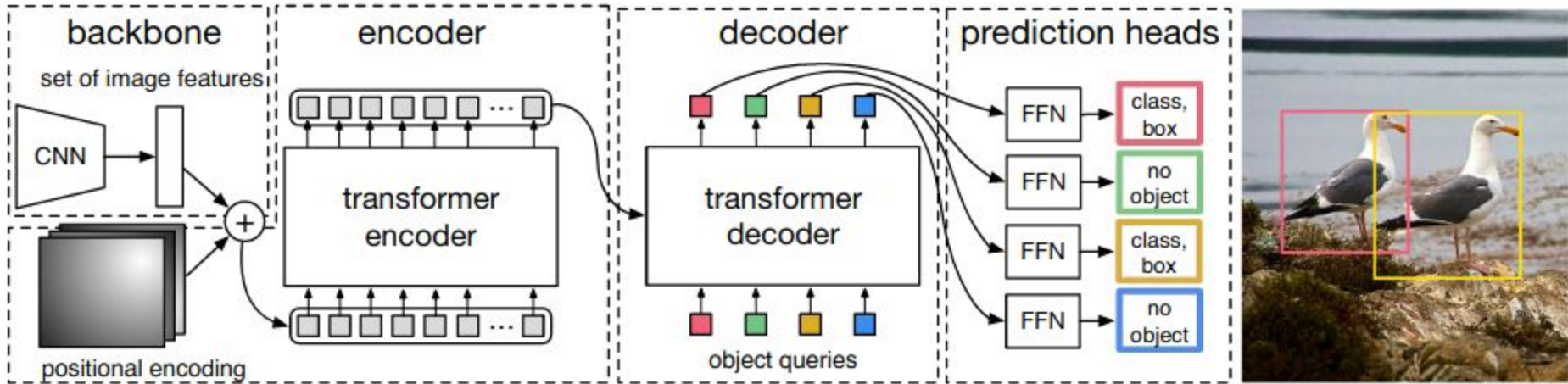


(d) One point for one identity

Detection with transformers - DETR

"End-to-End Object Detection with Transformers"

<https://arxiv.org/pdf/2005.12872.pdf>



Tracking with transformers - TrackFormer

"TrackFormer: Multi-Object Tracking with Transformers"

<https://arxiv.org/pdf/2101.02702.pdf>

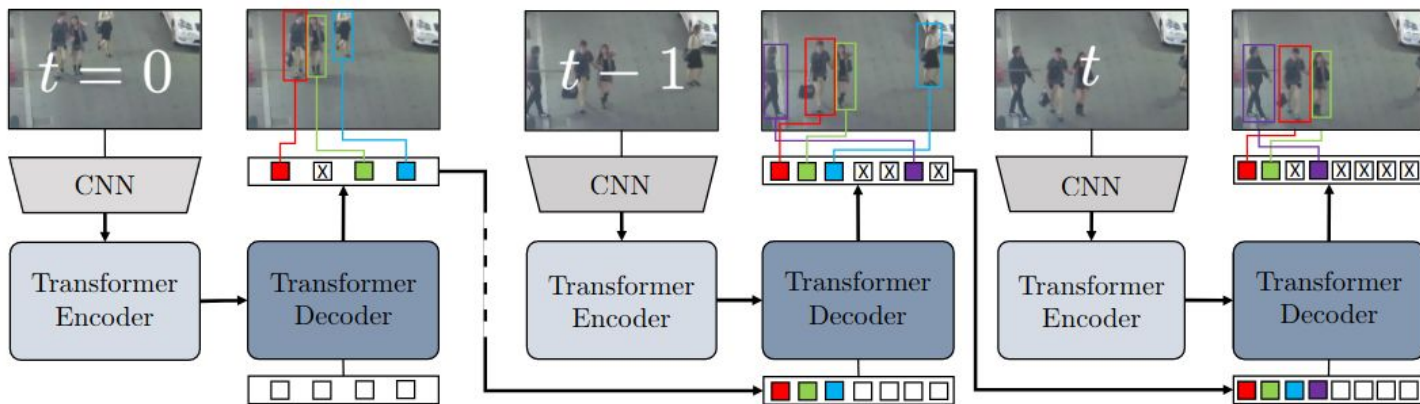


Figure 2. **TrackFormer** casts multi-object tracking as a set prediction problem performing joint detection and **tracking-by-attention**. The architecture consists of a CNN for image feature extraction, a Transformer [51] encoder for image feature encoding and a Transformer decoder which applies self- and encoder-decoder attention to produce output embeddings with bounding box and class information. At frame $t = 0$, the decoder transforms N_{object} object queries (white) to output embeddings either initializing new autoregressive **track queries** or predicting the background class (crossed). On subsequent frames, the decoder processes the joint set of $N_{\text{object}} + N_{\text{track}}$ queries to follow or remove (blue) existing tracks as well as initialize new tracks (purple).

Tracking with transformers - TrackFormer

Nice solution naturally merging detection and data association

Generally very good performance

Yet difficult to train, a lot of data required (MOT datasets are not sufficient)

Current trends

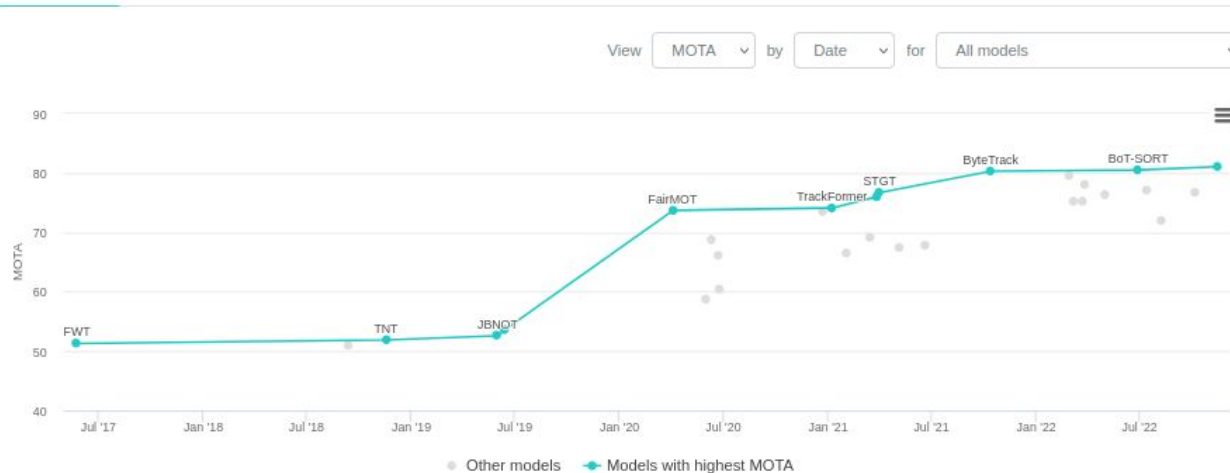
<https://paperswithcode.com/sota/multi-object-tracking-on-mot17>

<https://paperswithcode.com/sota/multi-object-tracking-on-mot20-1>

Let us see what are the approaches currently leading on the known benchmarks...

Multi-Object Tracking on MOT17

Leaderboard Dataset



Filter: untagged

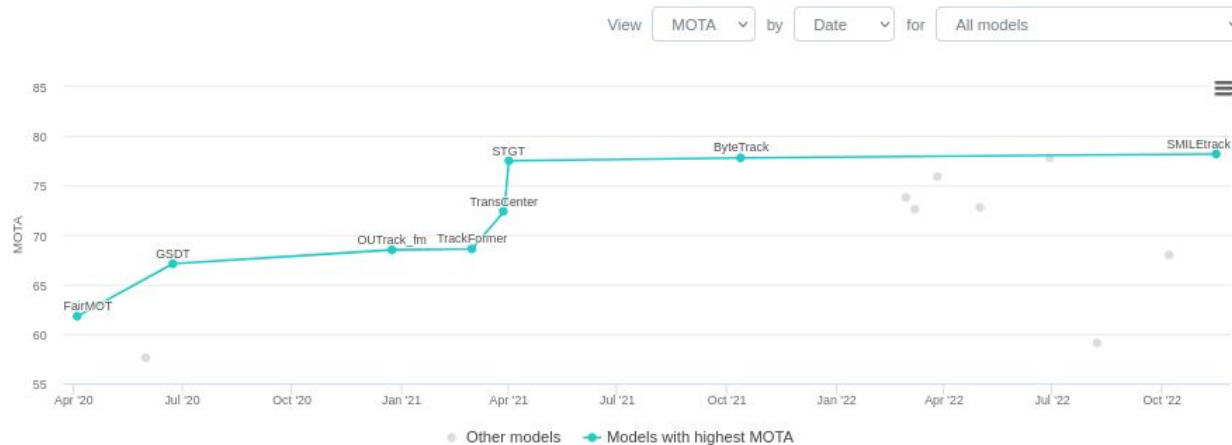
Edit Leaderboard

Rank	Model	HOTA	MOTA↑	IDF1	Extra Training Data	Paper	Code	Result	Year	Tags
1	SMILEtrack	65.24	81.06	80.5	✓	SMILEtrack: Similarity Learning for Multiple Object Tracking	Code	Result	2022	
2	BoT-SORT	65.0	80.5	80.2	✓	BoT-SORT: Robust Associations Multi-Pedestrian Tracking	Code	Result	2022	
3	ByteTrack		80.3	77.3	✓	ByteTrack: Multi-Object Tracking by Associating Every Detection Box	Code	Result	2021	
4	StrongSORT	64.4	79.6	79.5	✓	StrongSORT: Make DeepSORT Great Again	Code	Result	2022	

Multi-Object Tracking on MOT20

Leaderboard

Dataset



Filter: untagged

Edit Leaderboard

Rank	Model	MOTA↑	IDF1	HOTA	Extra Training Data	Paper	Code	Result	Year	Tags
1	SMILEtrack	78.19	77.53	63.43	✓	SMILEtrack: SiMilarity LEarning for Multiple Object Tracking	Code	Result	2022	
2	BoT-SORT	77.8	77.5	63.3	✓	BoT-SORT: Robust Associations Multi-Pedestrian Tracking	Code	Result	2022	
3	ByteTrack	77.8	75.2	61.3	✓	ByteTrack: Multi-Object Tracking by Associating Every Detection Box	Code	Result	2021	
4	STGT	77.5	75.2		✓	TransMOT: Spatial-Temporal Graph Transformer for Multiple Object Tracking		Result	2021	

Thus Tracking by detection!

Very good detections needed

Bipartite matching:

- define distance between boxes, e.g. IoU, pixel distance, reID -> **good features extracted and distances defined**
- solve the unique matching, e.g. with the Hungarian algorithm -> cannot change much here...

Exemplary approach with state-of-the-art results

"Hard to Track Objects with Irregular Motions and Similar Appearances? Make It Easier by Buffering the Matching Space"

<https://arxiv.org/pdf/2211.14317.pdf>

"Hard to Track Objects with Irregular Motions and Similar Appearances? Make It Easier by Buffering the Matching Space"

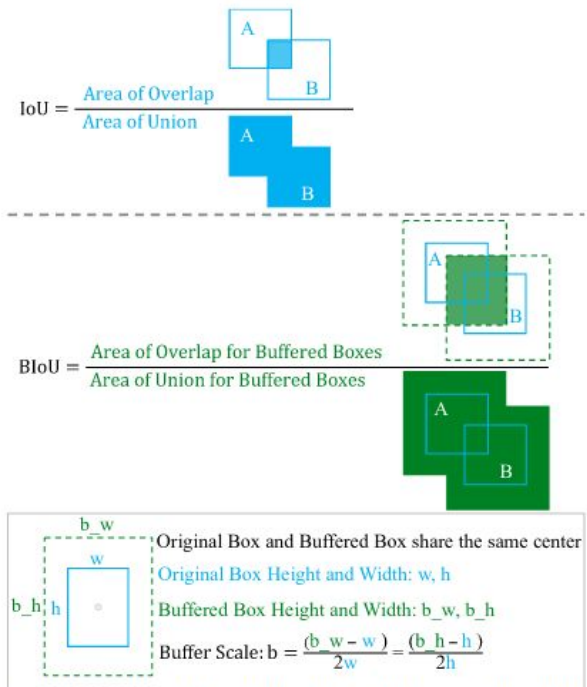


Fig. 2: Illustration of how Buffered IoU (BIOU) is calculated. Our BIOU adds a buffer that is proportional to the original bounding box. It does not change the location center, scale ratio, and shape of the original bounding boxes but expands the original matching space.

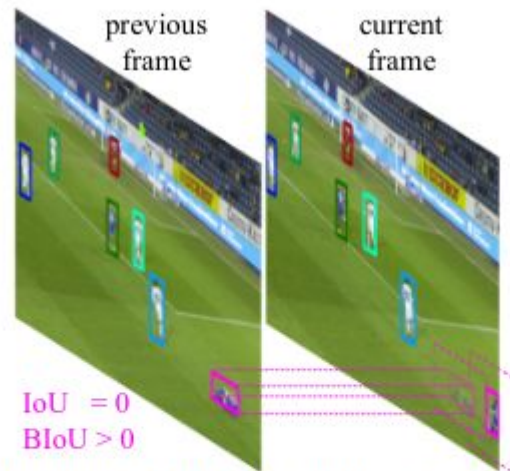


Fig. 3: An illustration of BIOU forms better cross-frame geometric consistency than IoU. The bounding box of an identical object shares the same color. The magenta object has no overlapping detections between adjacent frames. Whether this is caused by the fast movement or incorrect motion estimation, our BIOU expands the matching space to reduce the miss matching.

Thus current trends:

Tracking by detection, with simple yet powerful ideas and improvements

Questions and answers