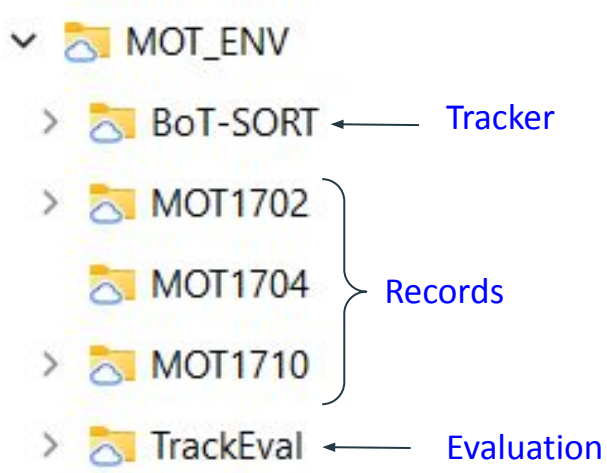
Improving Anchor Box Configuration

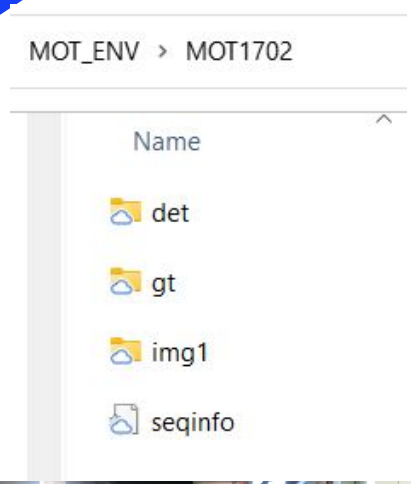
As a general rule, you should ask yourself the following questions about your dataset before diving into training your model:

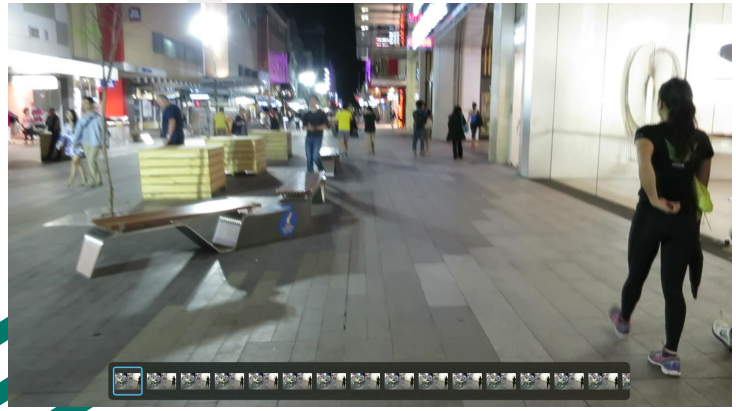
1. What is the smallest size box I want to be able to detect?
2. What is the largest size box I want to be able to detect?
3. What are the shapes the box can take? For example, a car detector might have short and wide anchor boxes as long as there is no chance of the car or the camera being turned on its side.

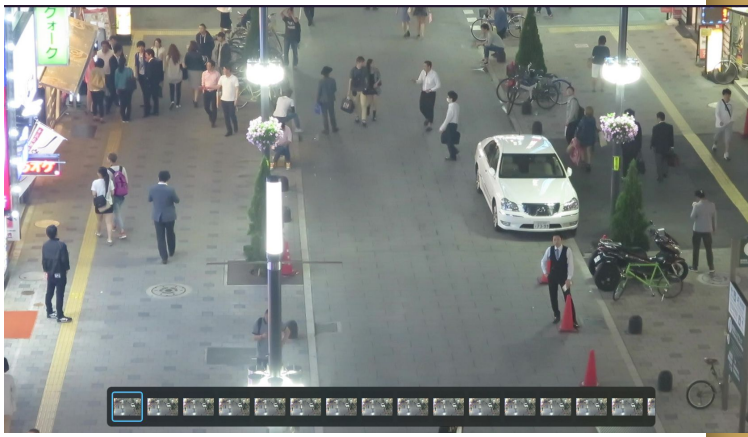
Exercise 4


MOT_ENV
BoT-SORT
Tracker
MOT1702
MOT1704
MOT1710
Records
TrackEval
Evaluation

Records







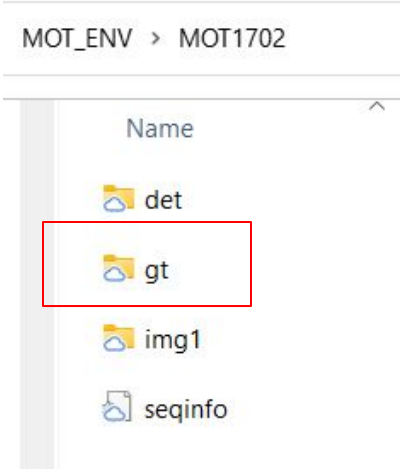
Data Format

The tracker file format should be the same as the ground truth file, which is a CSV text-file containing one object instance per line. Each line must contain 10 values:

```
<frame>,
<id>,
<bb_left>,
<bb_top>,
<bb_width>,
<bb_height>,
<conf>,
<x>,
<y>,
<z>
```

All frame numbers, target IDs and bounding boxes are 1-based. Here is an example:

```
1, 3, 794.27, 247.59, 71.245, 174.88, -1, -1, -1, -1
1, 6, 1648.1, 119.61, 66.504, 163.24, -1, -1, -1, -1
1, 8, 875.49, 399.98, 95.303, 233.93, -1, -1, -1, -1
...
```



Original https://motchallenge.net/results/MOT17/

Ex4 https://bj.bcebos.com/v1/paddledet/data/mot/

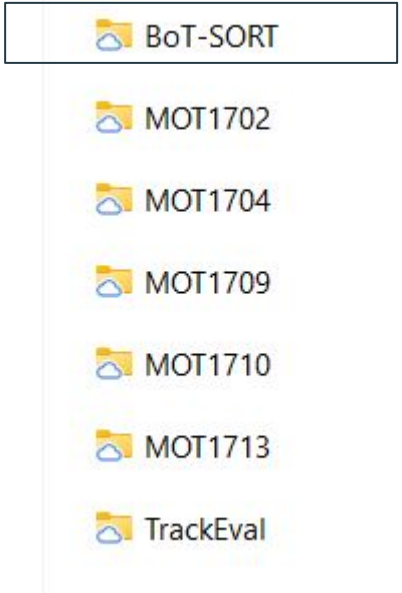
https://bj.bcebos.com/v1/paddledet/data/mot/MOT17.zip

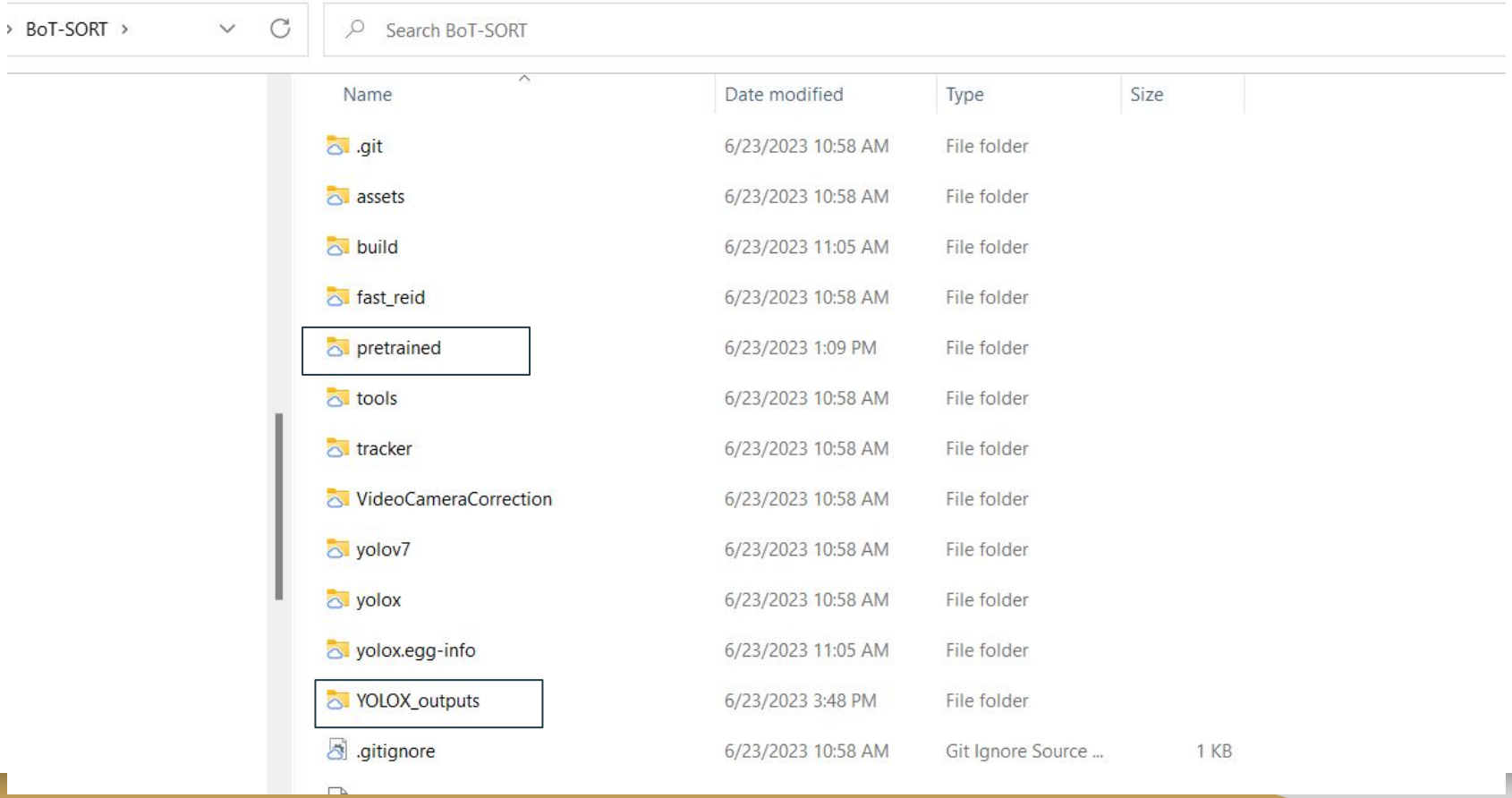
BoT-SORT
Tracker

paper: https://arxiv.org/abs/2206.14651

Official: https://github.com/NirAharon/BoT-SORT

Ex4 (colab) https://github.com/orfaig/BoT-SORT.git




BoT-SORT
Search BoT-SORT
Name
Date modified
Type
Size
.git 6/23/2023 10:58 AM File folder
assets 6/23/2023 10:58 AM File folder
build 6/23/2023 11:05 AM File folder
fast_reid 6/23/2023 10:58 AM File folder
pretrained 6/23/2023 1:09 PM File folder
tools 6/23/2023 10:58 AM File folder
tracker 6/23/2023 10:58 AM File folder
VideoCameraCorrection 6/23/2023 10:58 AM File folder
yolov7 6/23/2023 10:58 AM File folder
yolox 6/23/2023 10:58 AM File folder
yolox.egg-info 6/23/2023 11:05 AM File folder
YOLOX_outputs 6/23/2023 3:48 PM File folder
.gitignore 6/23/2023 10:58 AM Git Ignore Source ... 1 KB

ReID

- We used the publicly available ByteTrack model zoo trained on MOT17, MOT20 and ablation study for YOLOX object detection.
- Ours trained ReID models can be downloaded from MOT17-SBS-S50, MOT20-SBS-S50.
- For multi-class MOT use YOLOX or YOLOv7 trained on COCO (or any custom weights).

https://github.com/NirAharon/BoT-SORT

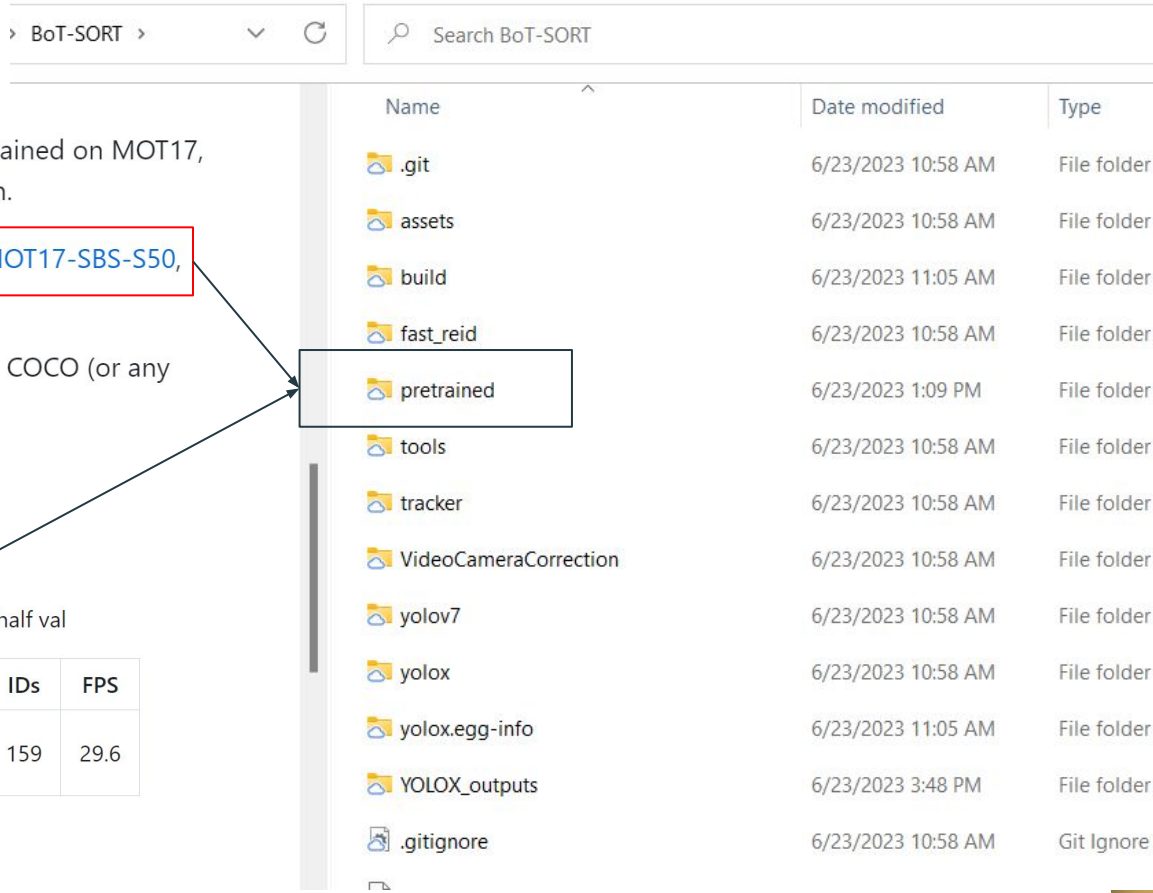
Yolox

Ablation model

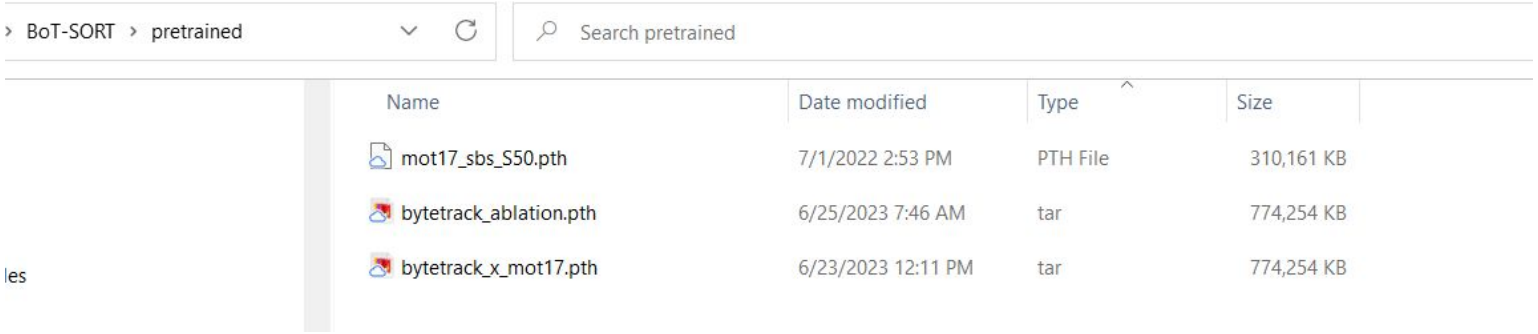
Train on CrowdHuman and MOT17 half train, evaluate on MOT17 half val

Model	MOTA	IDF1	IDs	FPS
ByteTrack_ablation [google], [baidu(code:eeo8)]	76.6	79.3	159	29.6

https://github.com/ifzhang/ByteTrack



BoT-SORT



**

The outputs after running the tracker show bounding box + ID, only when there is a detection for these objects

Official: https://github.com/JonathonLuiten/TrackEval

Ex4 (colab) https://github.com/orfaig/TrackEval

Readme: https://github.com/JonathonLuiten/TrackEval/blob/master/docs/MOTChallenge-Official/Readme.md

Metric Family	Sub metrics	Paper	Code	Notes
HOTA metrics	HOTA, DetA, AssA, LocA, DetPr, DetRe, AssPr, AssRe	paper	code	Recommended tracking metric
CLEARMOT metrics	MOTA, MOTP, MT, ML, Frag, etc.	paper	code	
Identity metrics	IDF1, IDP, IDR	paper	code	
VACE metrics	ATA, SFDA	paper	code	
Track mAP metrics	Track mAP	paper	code	Requires confidence scores
J & F metrics	J&F, J, F	paper	code	Only for Seg Masks
ID Euclidean	ID Euclidean	paper	code	

MOT evaluation report!

HOTA: BoTSORT-pedestrian	HOTA	DetA	AssA	DetRe	DetPr	AssRe	AssPr	LocA	OWTA	HOTA(0)	LocA(0)	HOTALocA(0)
MOT16-05	66.659	66.648	48.801	71.85	85.84	55.458	71.421	87.058	59.055	66.741	85.511	55.751
MOT16-11	80.567	90.885	89.956	95.506	94.112	92.855	95.712	92.779	91.71	97.194	92.176	89.589
MOT16-15	68.679	69.651	67.916	74.807	80.952	75.288	79.555	85.851	71.271	87.599	79.97	69.895
COMBINED	79.84	81.055	79.075	85.572	89.874	85.105	87.896	90.518	82.127	89.08	88.497	78.855

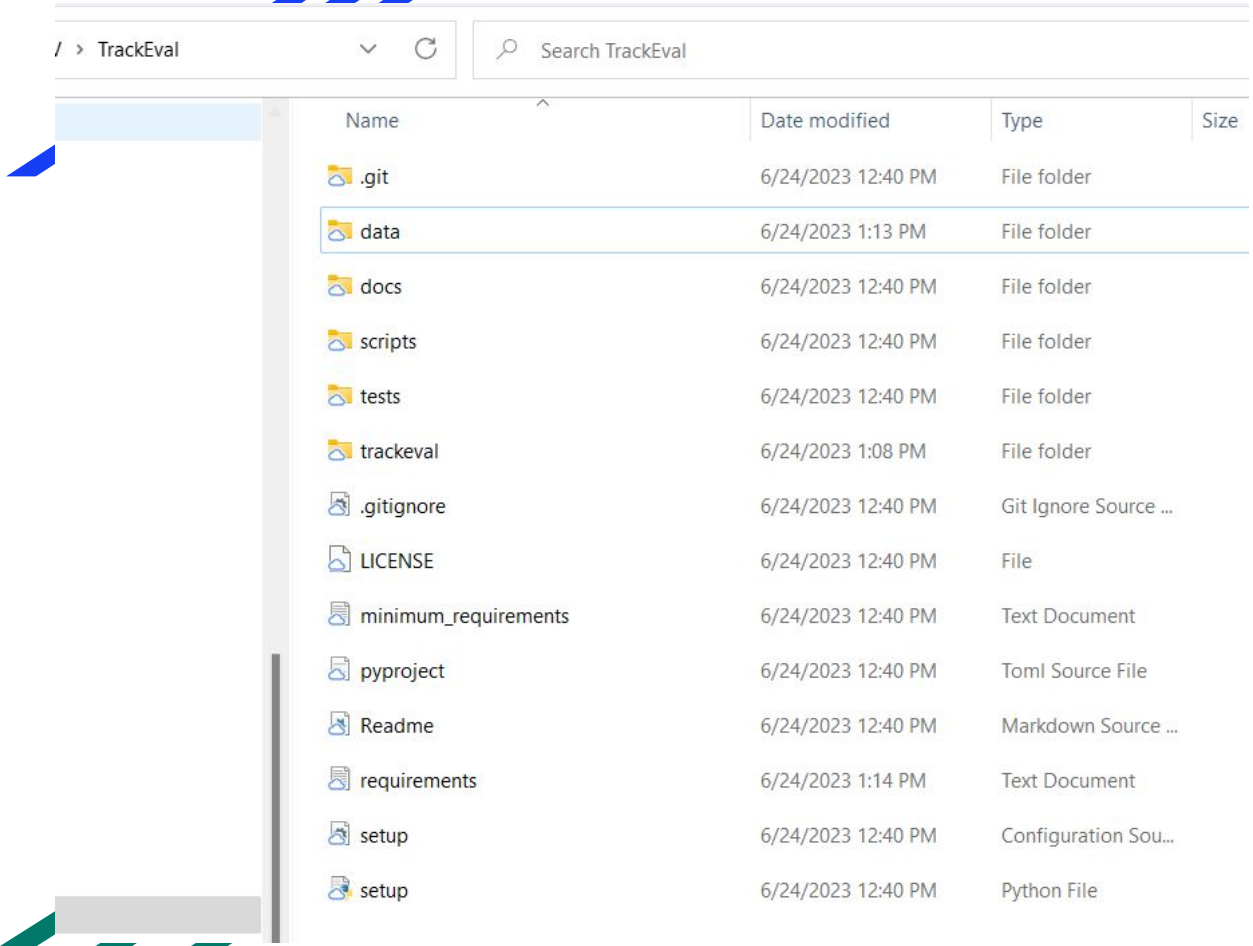
CLEAR: BoTSORT-pedestrian	MOTA	MOTP	MODA	CLR_Re	CLR_Pr	MTR	PTR	MLR	sMOTA	CLR_TP	CLR_FN	CLR_FP	IDSW	MT	PT
MOT16-05	76.949	84.859	77.847	81.761	95.452	67.925	22.642	9.454	64.57	8105	1808	588	89	56	12
MOT16-11	85.528	92.525	98.565	98.898	99.464	98.551	0	1.4495	90.756	25959	267	129	9	68	0
MOT16-15	81.859	82.057	86.209	89.521	96.654	77.778	19.444	2.7778	69.786	5511	655	185	22	28	7
COMBINED	91.189	89.241	91.488	95.259	98.156	85.544	12.055	4.4511	81.157	57575	2715	705	120	152	19

Identity: BoTSORT-pedestrian	IDF1	IDR	IDP	IDTP	IDFN	IDFP
MOT16-05	66.652	58.146	67.868	5764	4149	2729
MOT16-11	89.115	96.858	97.595	25460	766	628
MOT16-15	85.807	82.56	89.52	4909	1057	587
COMBINED	86.559	85.152	89.642	54155	5952	5944

VACE: BoTSORT-pedestrian	SFDA	ATA
MOT16-05	76.705	40.211
MOT16-11	91.668	91.552
MOT16-15	77.484	68.82
COMBINED	85.745	66.406

Count: BoTSORT-pedestrian	Dets	GT_Dets	IDs	GT_IDs
MOT16-05	8495	9915	90	55
MOT16-11	24088	24226	72	69
MOT16-15	5496	5946	47	56
COMBINED	58077	40085	209	158

- Compare the results at a high level
- Why do you think there is a difference in the results?
- Where is the tracker found the most has the best performance and why?
  (dynamic/static scene, occlusions,trunked objects,` detection performance, crowded scene..)
  Please add 1-2 examples.


/ › TrackEval
Search TrackEval
Name
Date modified
Type
Size
.git 6/24/2023 12:40 PM File folder
data 6/24/2023 1:13 PM File folder
docs 6/24/2023 12:40 PM File folder
scripts 6/24/2023 12:40 PM File folder
tests 6/24/2023 12:40 PM File folder
trackeval 6/24/2023 1:08 PM File folder
.gitignore 6/24/2023 12:40 PM Git Ignore Source ...
LICENSE 6/24/2023 12:40 PM File
minimum_requirements 6/24/2023 12:40 PM Text Document
pyproject 6/24/2023 12:40 PM Toml Source File
Readme 6/24/2023 12:40 PM Markdown Source ...
requirements 6/24/2023 1:14 PM Text Document
setup 6/24/2023 12:40 PM Configuration Sou...
setup 6/24/2023 12:40 PM Python File

To run the evaluation for your method please run the script at `TrackEval/scripts/run_mot_challenge.py`.

Some of the basic arguments are described below. For more arguments, please see the script itself.

`BENCHMARK` : Name of the benchmark, e.g. MOT15, MO16, MOT17 or MOT20 (default : MOT17)

`SPLIT_TO_EVAL` : Data split on which to evalute e.g. train, test (default : train)

`TRACKERS_TO_EVAL` : List of tracker names for which you wish to run evaluation. e.g. MPNTrack (default: all trackers in tracker folder)

`METRICS` : List of metric families which you wish to compute. e.g. HOTA CLEAR Identity VACE (default: HOTA CLEAR Identity)

`USE_PARALLEL` : Whether to run evaluation in parallel on multiple cores. (default: False)

`NUM_PARALLEL_CORES` : Number of cores to use when running in parallel. (default: 8)

An example is below (this will work on the supplied example data above):

```
!python scripts/run_mot_challenge.py --SPLIT_TO_EVAL 'test' --BENCHMARK 'MOT17' --GT_FOLDER
'/content/gdrive/MyDrive/MOT_ENV/TrackEval/data/gt' --TRACKERS_FOLDER
'/content/gdrive/MyDrive/MOT_ENV/TrackEval/data/trackers' --USE_PARALLEL False --METRICS HOTA CLEAR Identity
VACE
```

Riddle

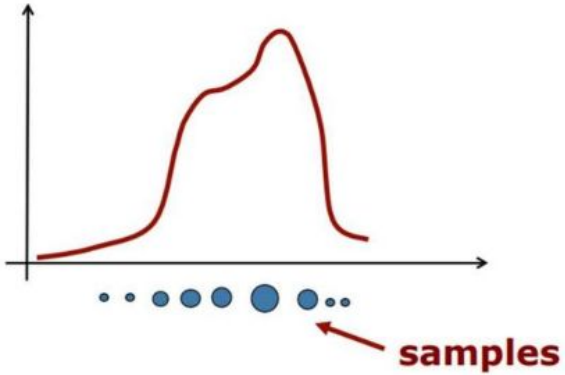
1.

I built a new autonomous tractor!
The robotic kit includes mono camera and odometer.

The tractor should be self-driving in sunflower fields.
I would like to use visual odometry mechanism for reducing robot pose drifts.

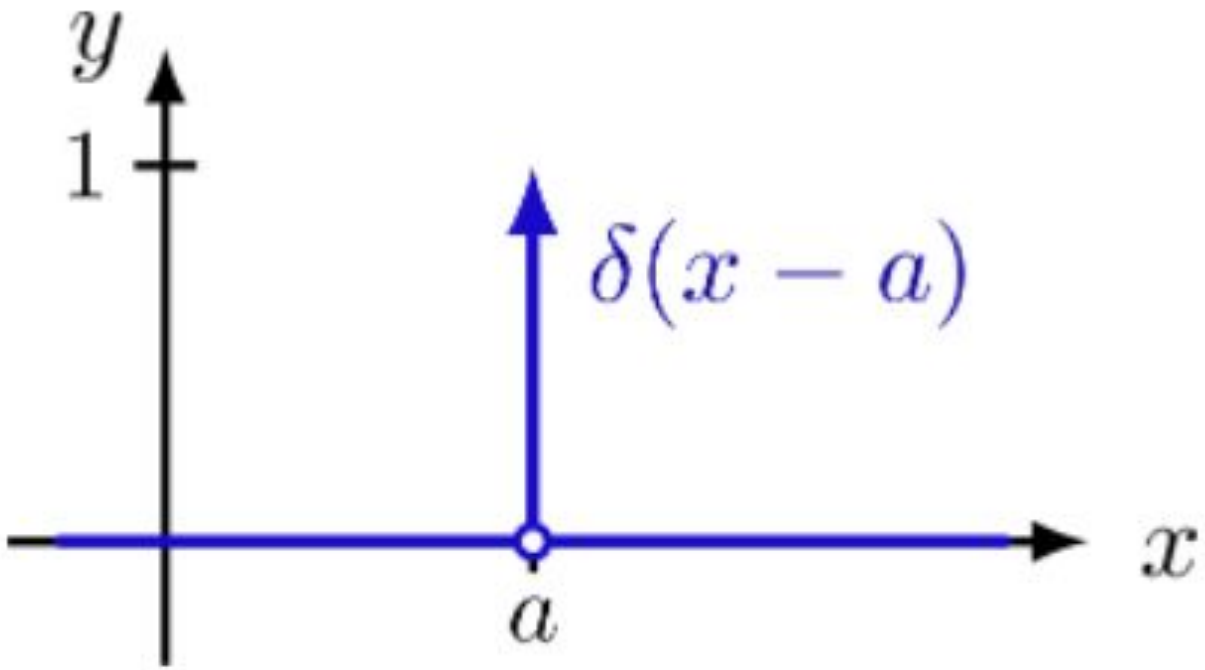
Which detector do you recommend to use for feature extraction (key points)?

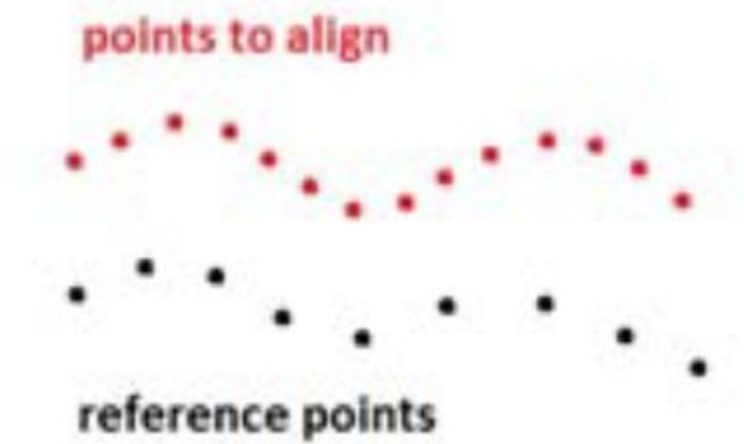
**Bear Flag Robotics- Autonomous tractor!



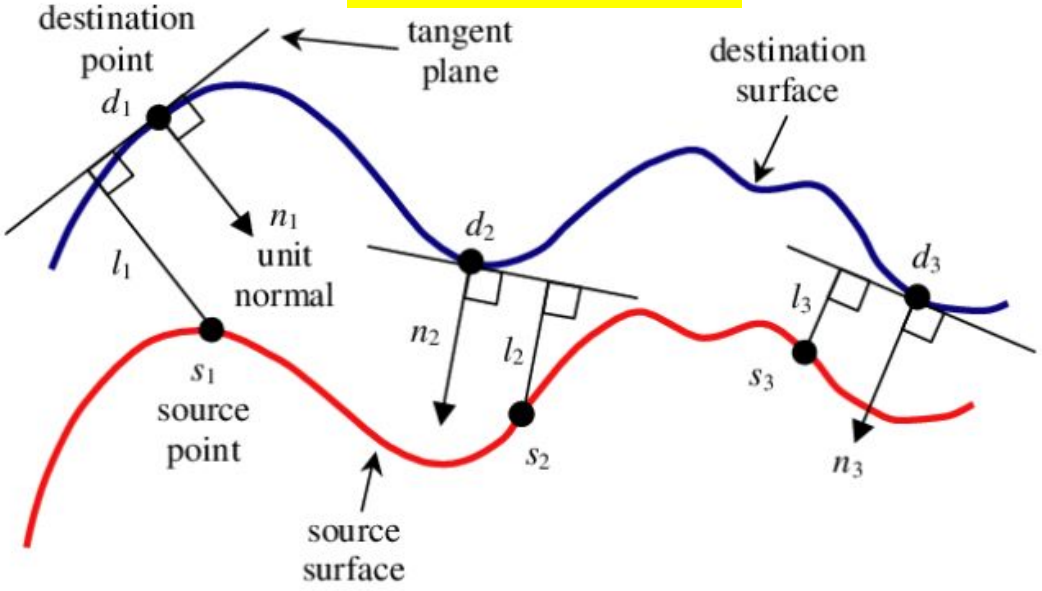
(for example)

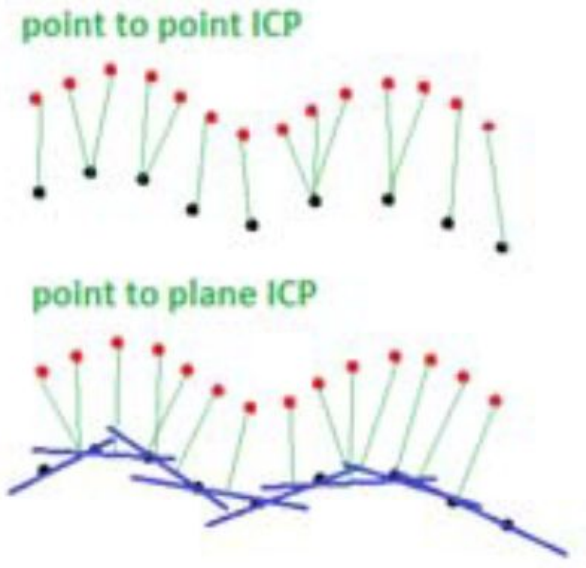
Which a familiar probability distribution is very hard/impossible to model based on the particle's mechanism?


y
1
δ(x − a)
x
a


points to align
reference points

Point to Plane!





Course Plan

1. Description of self-driving car system components, infrastructure.
2. Robot Mapping, Probabilistic occupancy mapping.
3. Semantic segmentation(2D/3D) , sensor fusion.
4. Kalman filter, Extended Kalman filter
5. Simultaneous localization and mapping (SLAM), EKF-SLAM
6. Particle filter , FAST 1.0, FAST 2.0
7. Point cloud registration (ICP). Pose Graph SLAM. LIO-SAM,GTSAM
8. Visual Odometry. ORB-SLAM
9. Obstacle Detection 2D
10. Obstacle Detection 3D, PointNet, PointPillars, PV-RCNN
11. Multi Object Tracking ,SORT, DeepSORT , BOT-SORT.
12. Autonomous vehicle lab- Live demo

Introduction

Robot mapping

localization and mapping

Perception

Live Demo!

+5 riddles!!


24-222-85


SAMSUNG


IAI
AUTONOMOUS LAB

Keep in touch!

Roy orfaig

Orfaig@gmail.com

https://www.linkedin.com/in/roy-orfaig-8975371a/

Prof Ben-Zion Bobrovsky

bobrov@tauex.tau.ac.il

https://www.linkedin.com/in/ben-zion-bobrovsky-062625143/

Thank you all for your listening