Robust Object Tracking for Inventory Monitoring

Ying Gu, Jingjie Guo, Aras Bayrakceken, Sarah El Beji, Alper Kagan Kayali and Benjamin Bangert

Mentors: M.Sc. Mathias Sundholm, M.Sc. Maximilian Schreil, M.Sc. Alexander Dolokov (PreciBake GmbH)

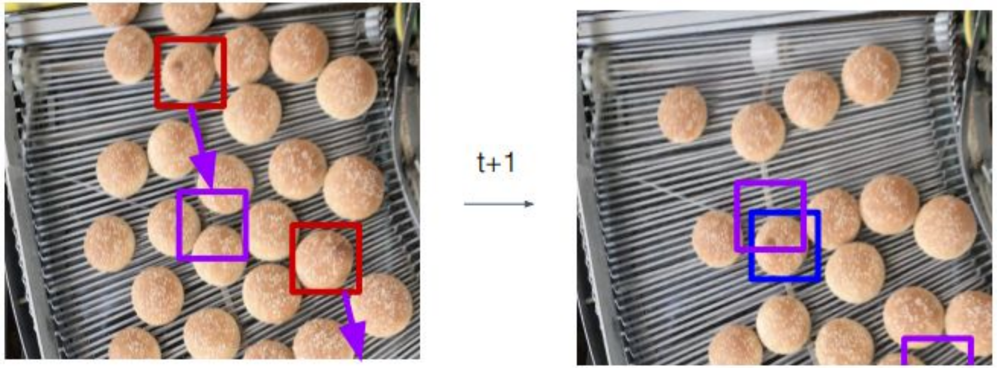
Project lead: Dr. Ricardo Acevedo Cabra (MDSI)

Supervisor: Prof. Dr. Massimo Fornasier (Board of Directors of MDSI)

Feb 2022

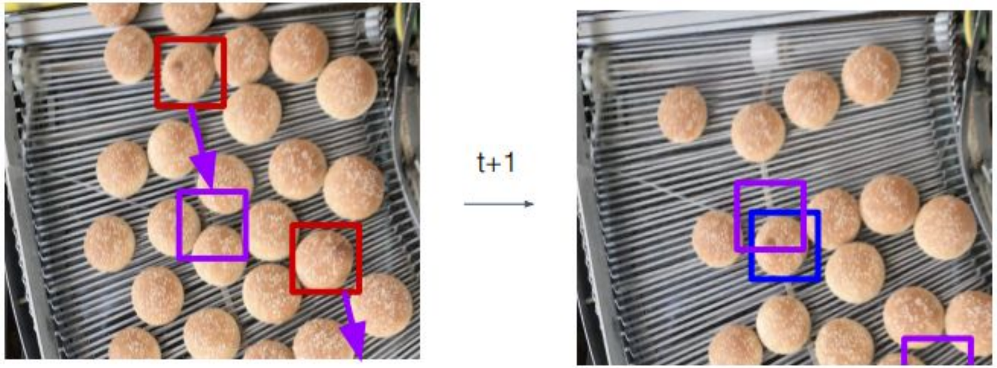
Object Tracking

- Estimating the trajectory of an object as it moves around a scene



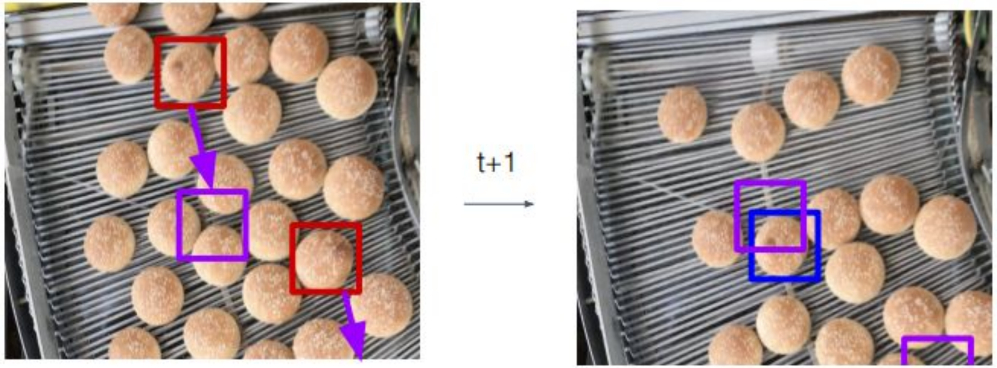
Object Tracking

- Estimating the trajectory of an object as it moves around a scene
- Use cases:
  - Surveillance
  - Vehicle Navigation
  - In our case: Inventory Monitoring
    - Inventory Size
    - Inventory Age
    - Stockout
    - Waste



Object Tracking

- Estimating the trajectory of an object as it moves around a scene
- Use cases:
    - Surveillance
    - Vehicle Navigation
    - In our case: Inventory Monitoring
- Challenges:
    - Noise
    - Complex motion
    - Object occlusion
    - Complex shapes



Object Tracking Datasets

- Issues:
  - Ambiguous ground truth
  - Different evaluation metrics ➜ Different results
  - Pre-defined test and training data

Fig. 5: The annotations include different classes of objects similar to the target class, a pedestrian in our case. We consider these special classes (distractor, reflection, static person and person on vehicle) to be so similar to the target class that a tracker should neither be penalized nor rewarded for tracking them in the sequence.

Object Tracking Datasets

- Issues:
  - Ambiguous ground truth
  - Different evaluation metrics ➜ Different results
  - Pre-defined test and training data
- Datasets:
  - PETS
  - KITTI
  - DETRAC
  - **MOTChallenge**

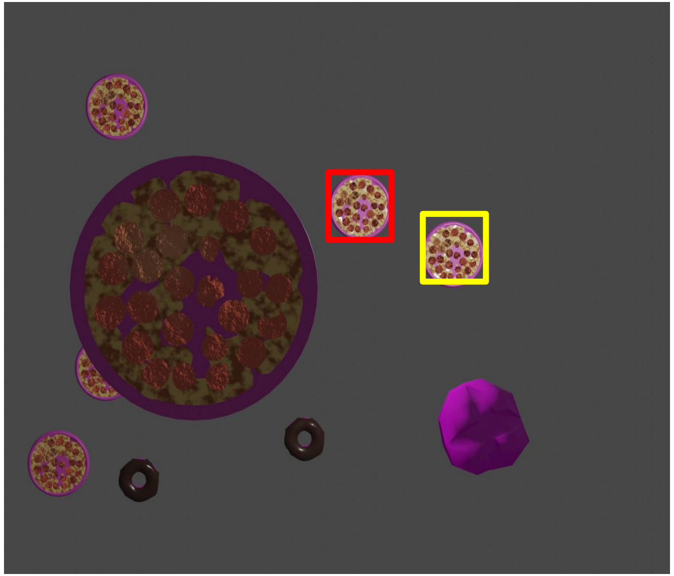
Fig. 5: The annotations include different classes of objects similar to the target class, a pedestrian in our case. We consider these special classes (distractor, reflection, static person and person on vehicle) to be so similar to the target class that a tracker should neither be penalized nor rewarded for tracking them in the sequence.

Simple Online and Realtime Tracking

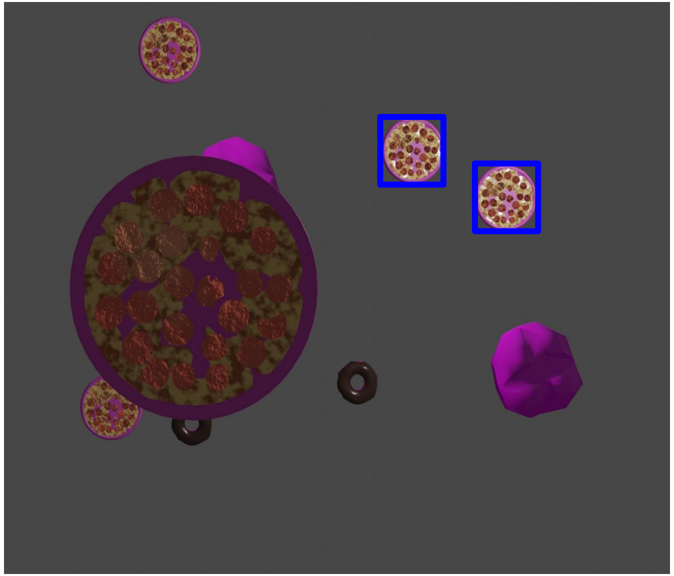
Detection

- Faster region CNN

cameraview at time t



cameraview at time t+1



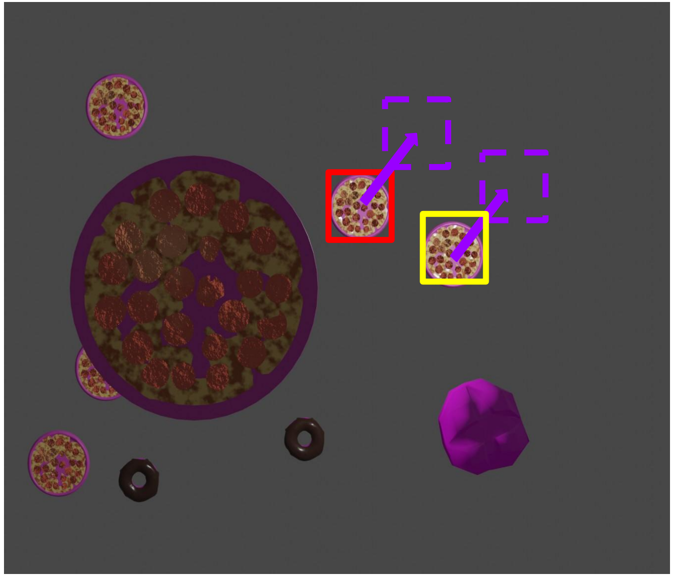
Bewley et al. Simple online and realtime tracking 2016

Simple Online and Realtime Tracking

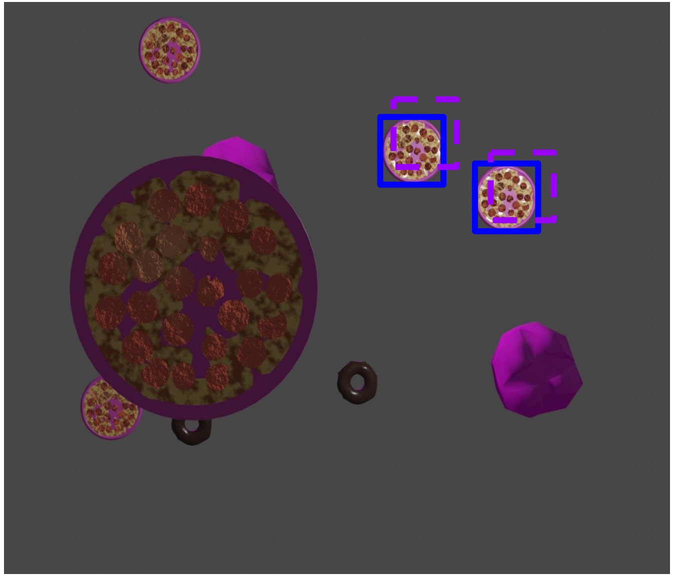
Estimation Model

- Linear velocity model
  - solved by Kalman filter framework

cameraview at time t



cameraview at time t+1



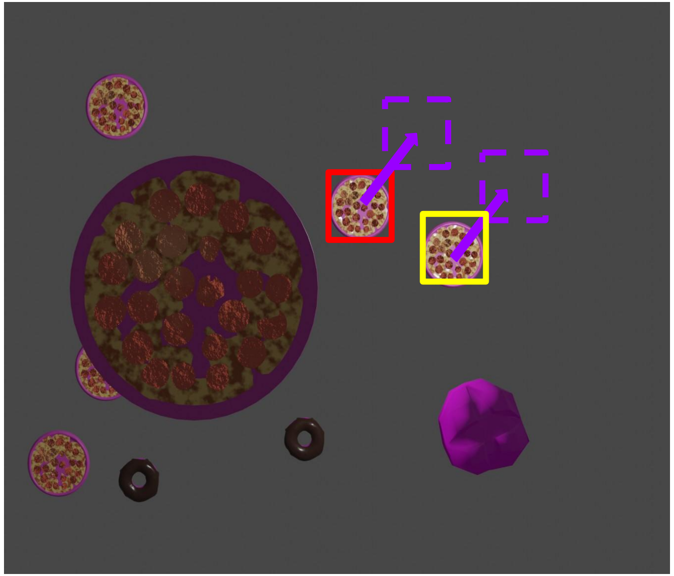
Bewley et al. Simple online and realtime tracking 2016

Simple Online and Realtime Tracking

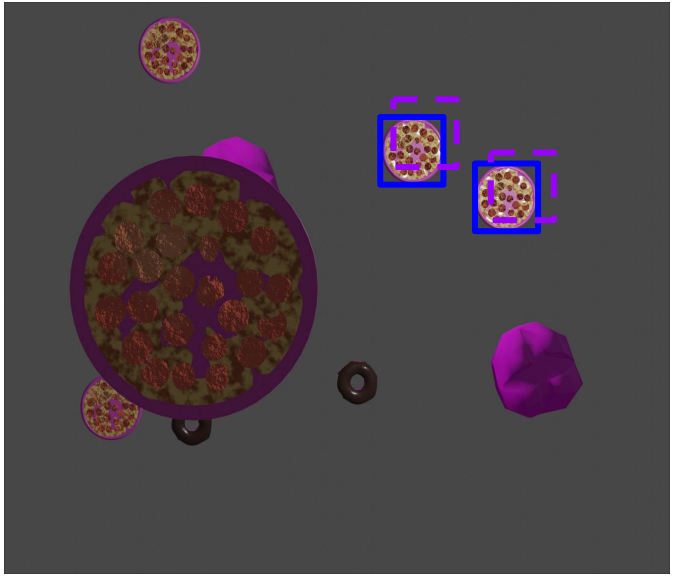
Data Association

- Hungarian Algorithm: solves bipartite matching problem
  - cost matrix IOU distance between detection and prediction

cameraview at time t



cameraview at time t+1



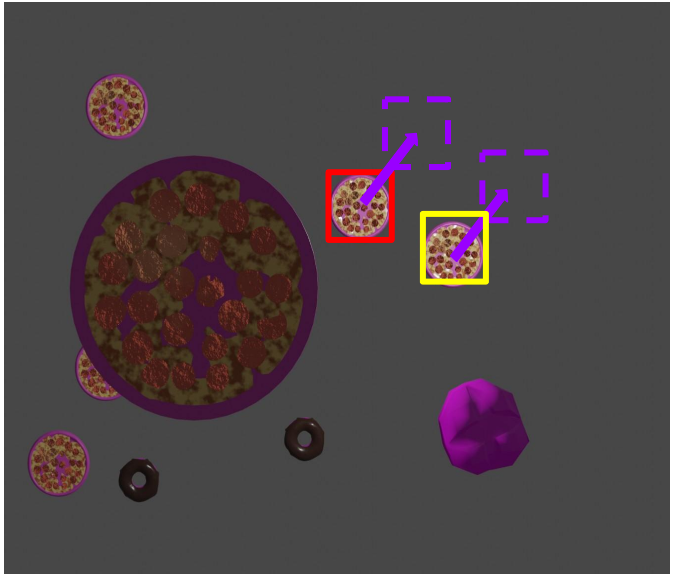
Bewley et al. Simple online and realtime tracking 2016

Simple Online and Realtime Tracking

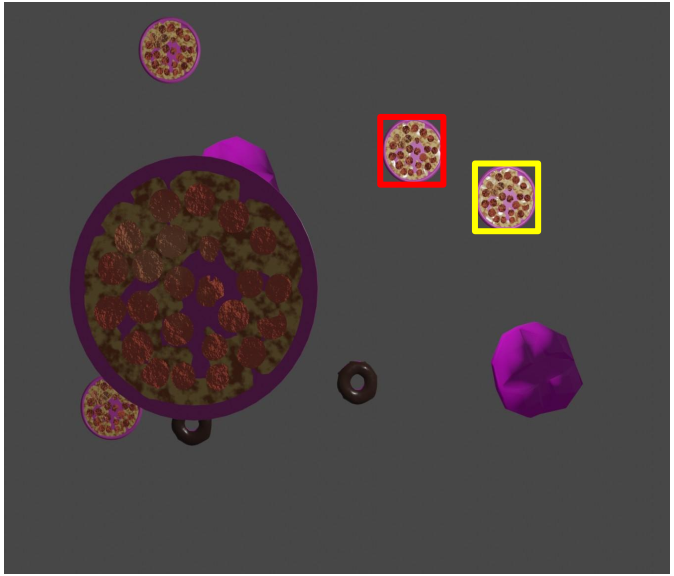
Data Association

- Hungarian Algorithm: solves bipartite matching problem
    - cost matrix IOU distance between detection and prediction

cameraview at time t



cameraview at time t+1



Bewley et al. Simple online and realtime tracking 2016

Simple Online and Realtime Tracking

Limitations

- Object reidentification (ReID)
  - Occlusion:
    - target is covered by occluding object
  - Reentering of objects to cameraview
    - object leaves camerview at time t, reenters at time t+n

- Fast moving objects wrt. fps
  - poor prediction of true dynamics

Bewley et al. Simple online and realtime tracking 2016

Simple Online and Realtime Tracking

Evaluate extend of limitataions

- Problem:
  - find suitable test data

- Solution:
  - simulate data using
  - advantages:
    - trigger modes c
    - flexibility in amount and quality of data
    - sustainable method - can be used for future experiments

Bewley et al. Simple online and realtime tracking 2016

Blender

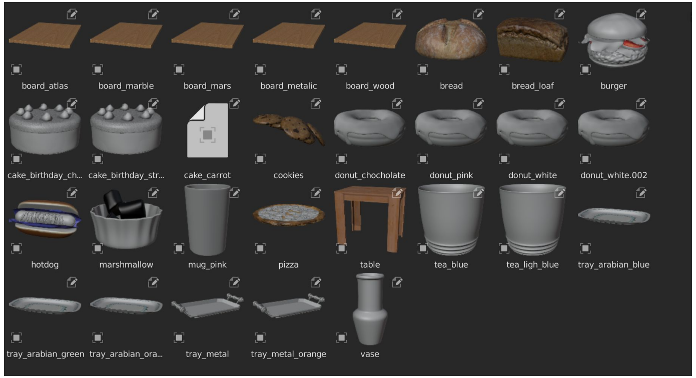
Pipeline

1. Create Scene
   a. simulate mode of failure from SORT
2. Extract Ground Truth Data
   a. bounding boxes (bbox)
   b. tracks
   c. correct bbox of occluded objects
3. Simulate Detector
   a. add noise to extracted ground truth data
      i. noise to bbox shape and position
      ii. remove detection
      iii. add false positive detection

Blender environment : Blender KIT

Partially free shared Library with an open community

- Advantages :
  - Pre-created models in food industry context
- Downsides :
  - Manually cleaned
  - Texture and join problems
  - Limited models

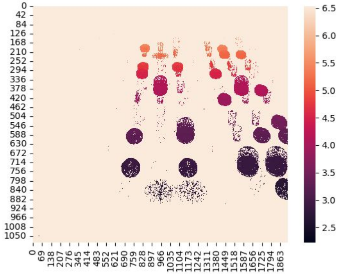


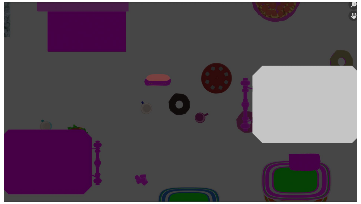
Asset Browser of 21 different objects

Use case

Rendering : From assets to tracks

Depth-z behaves weirdly for big objects





Texture unstable
Transformations changed

Wrong rendering for heavy scenes

Fast sanity check : No problem for Eevee renderer

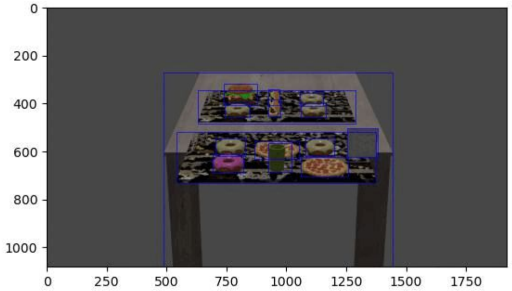
Common asset.blend file

rendered scenes

Partial occluded bounding boxes

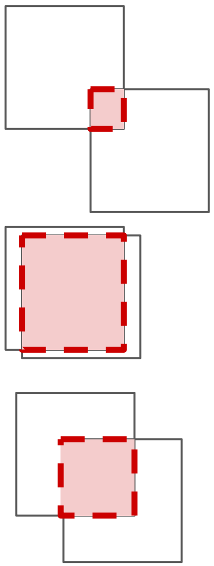
Scripts with/without failure modes

Tracks and Bounding for all objects



Bounding Boxes for occlusion

Calculate total occlusion area
Compare to empirical thresholds

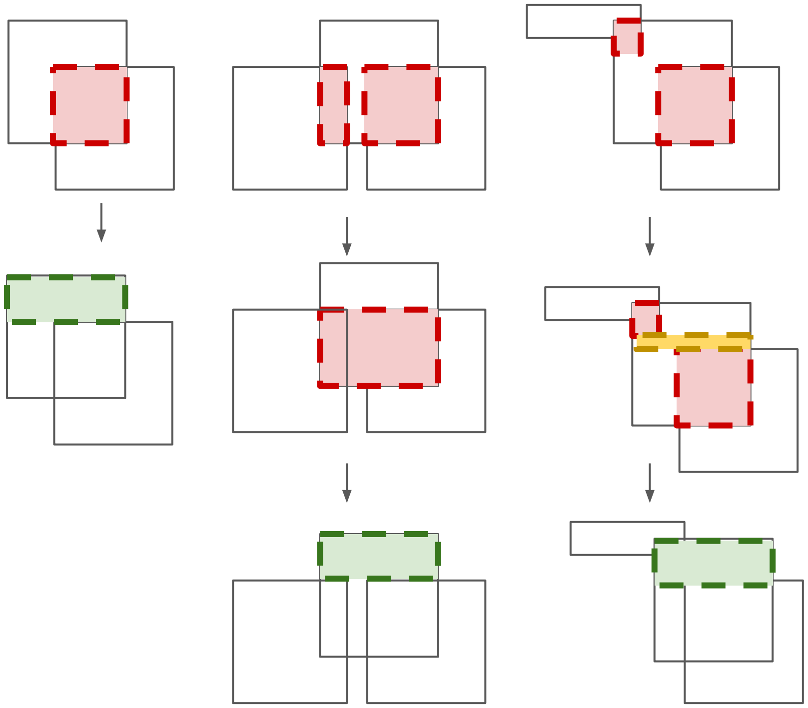


No occlusion (.<0.2)

Total occlusion (.>0.6)

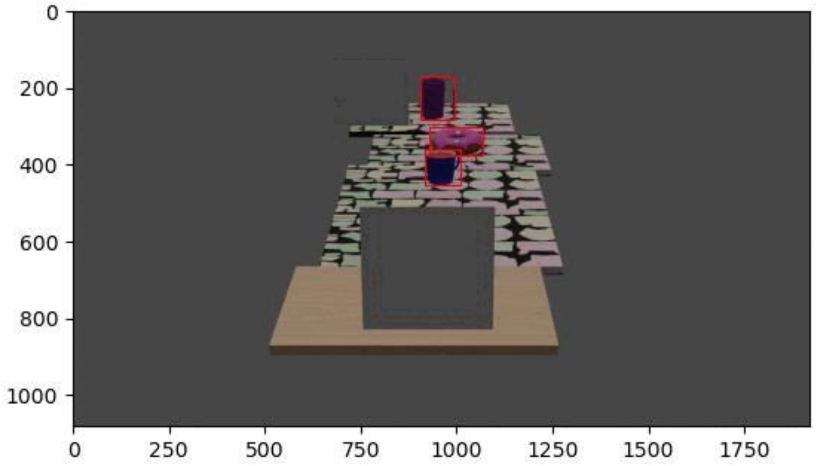
Partial occlusion (0.2 <.<0.6)

Partial Occlusion

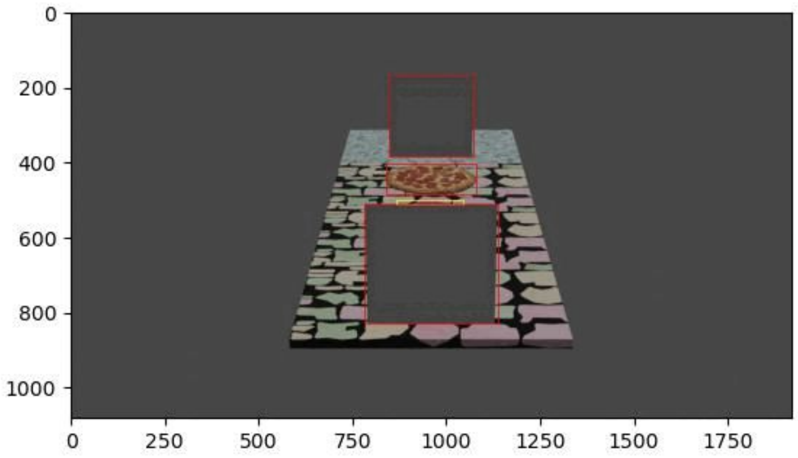


Problem is for more than 2 occluders

Results



Partial occlusion (0.3 <.<0.7)

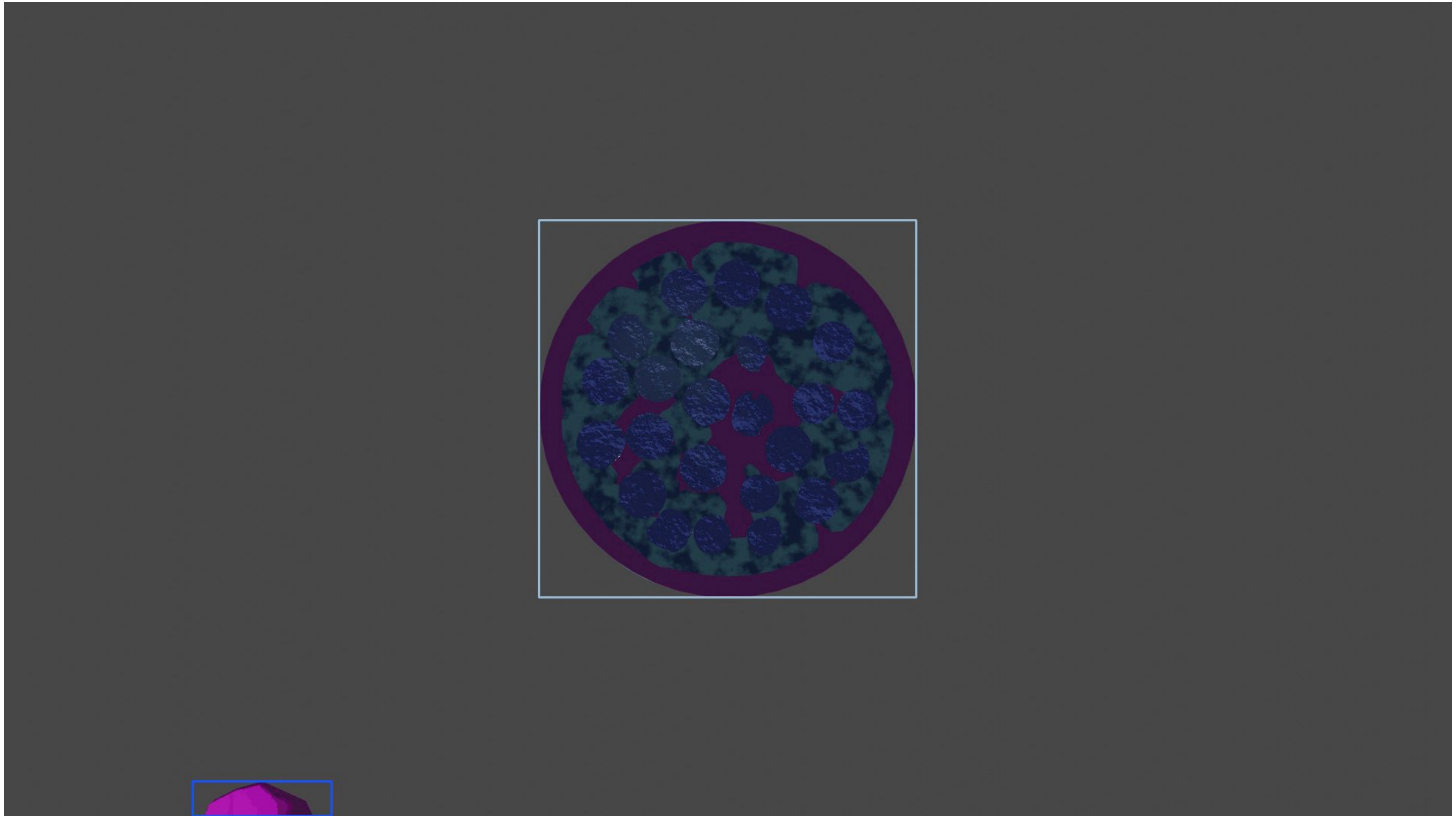


Partial occlusion (0 <.<1)

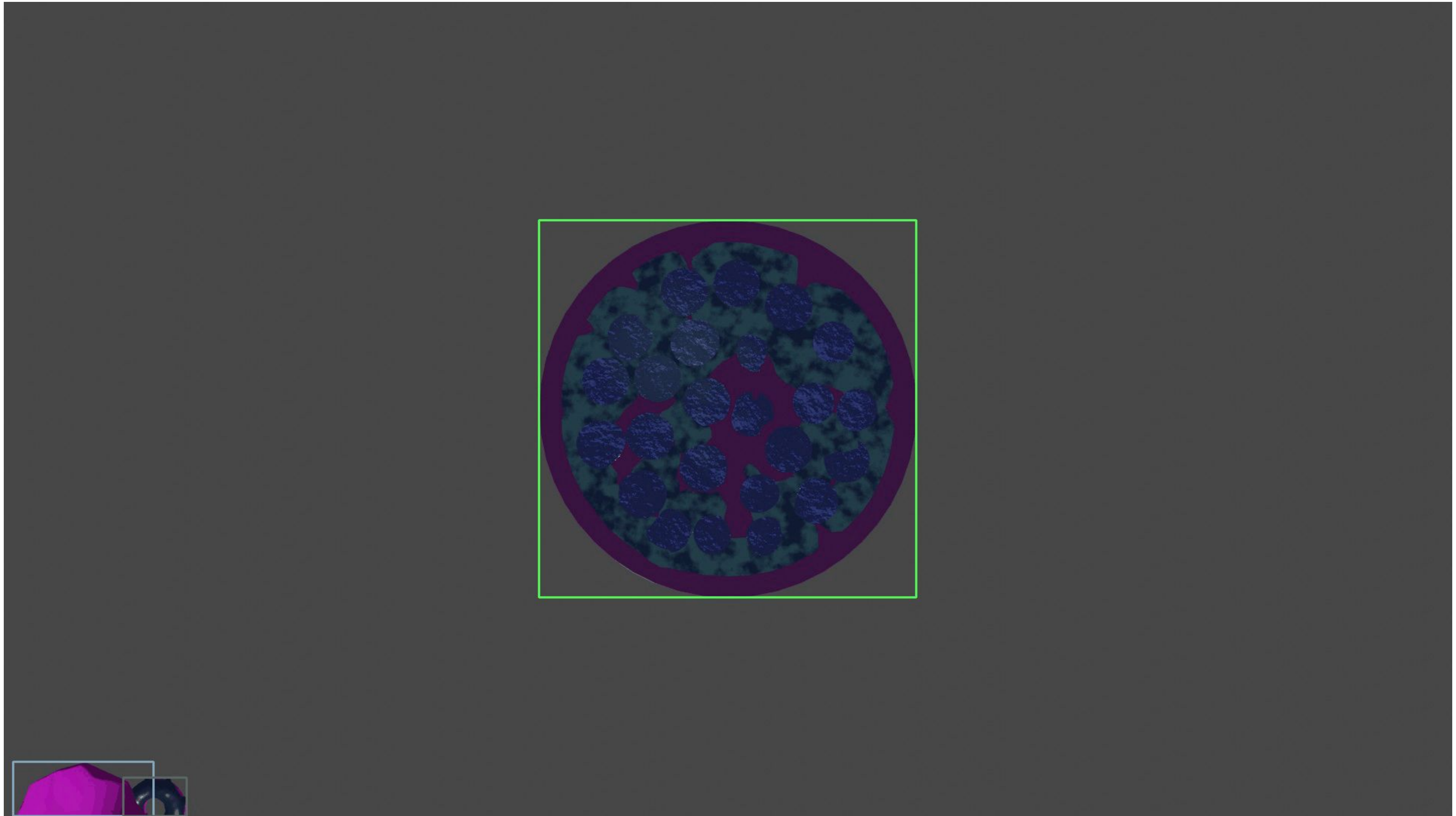
Globalization and Automation of the Scenes

- Divided the automation into several parts:
  - Ground truth extraction compatible with SORT and other algorithms
  - Randomizing the features of the objects in the scene
- Continued with Different Failure Cases

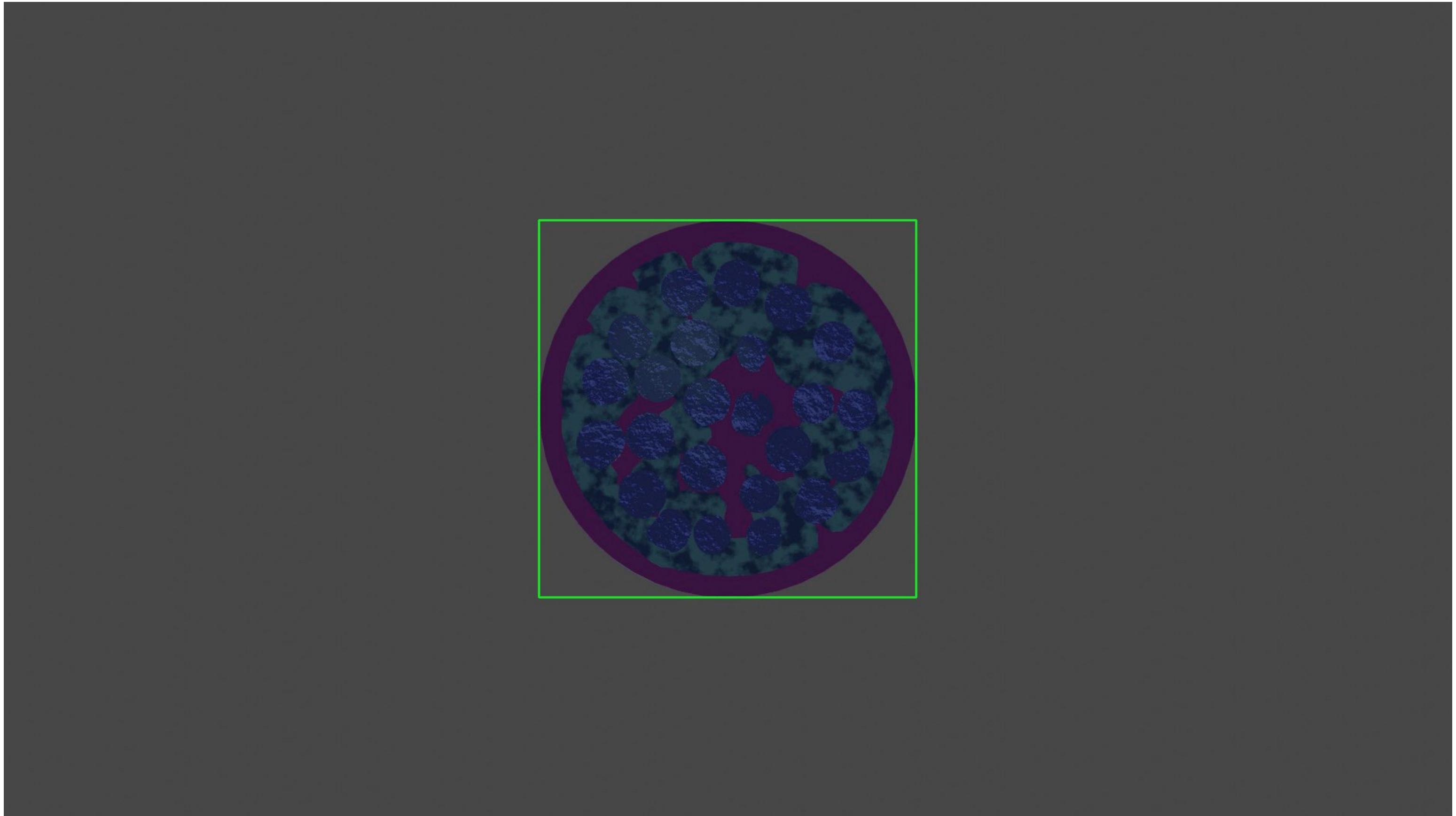
Regular Occlusion Scene with a Familiar Object

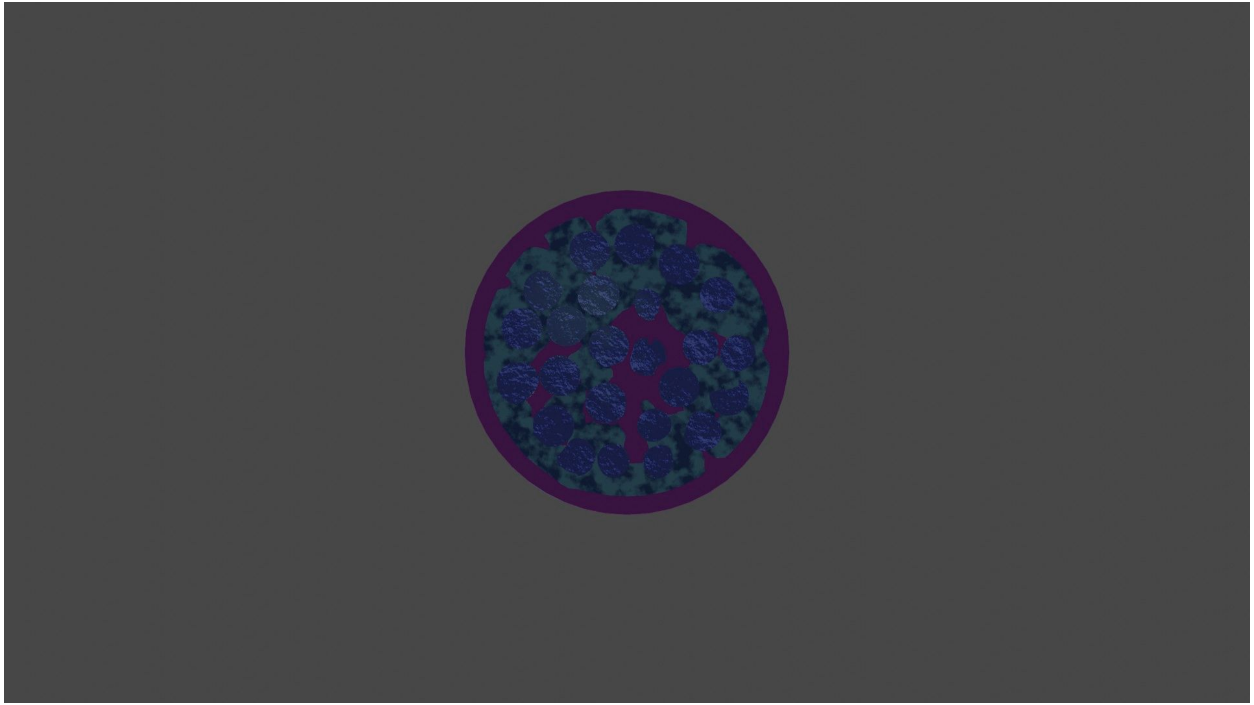


Occlusion With FPS Drop

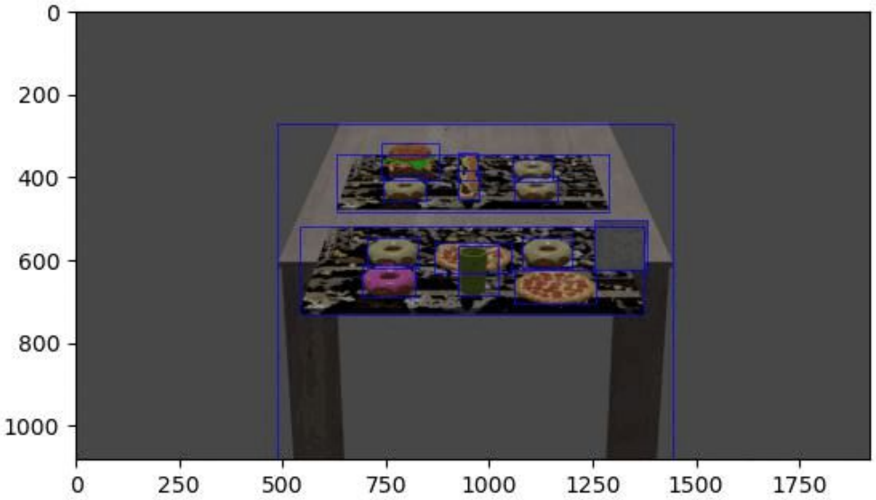


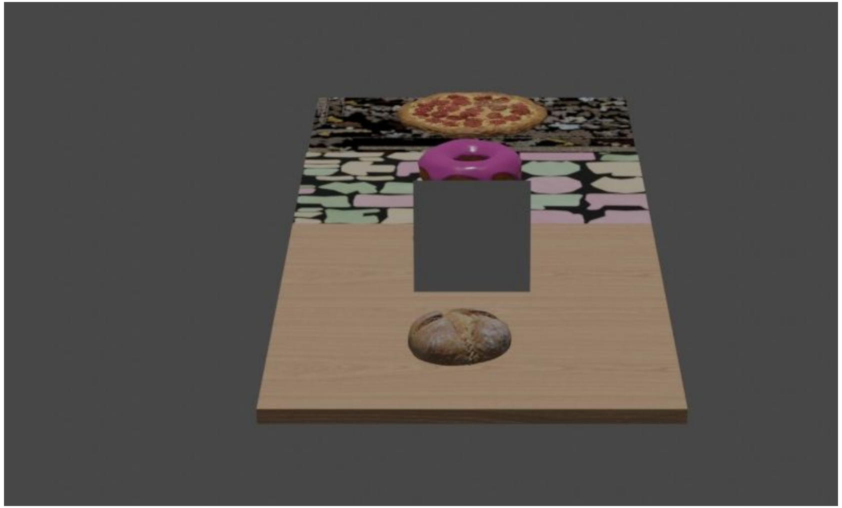
Occlusion with Lag



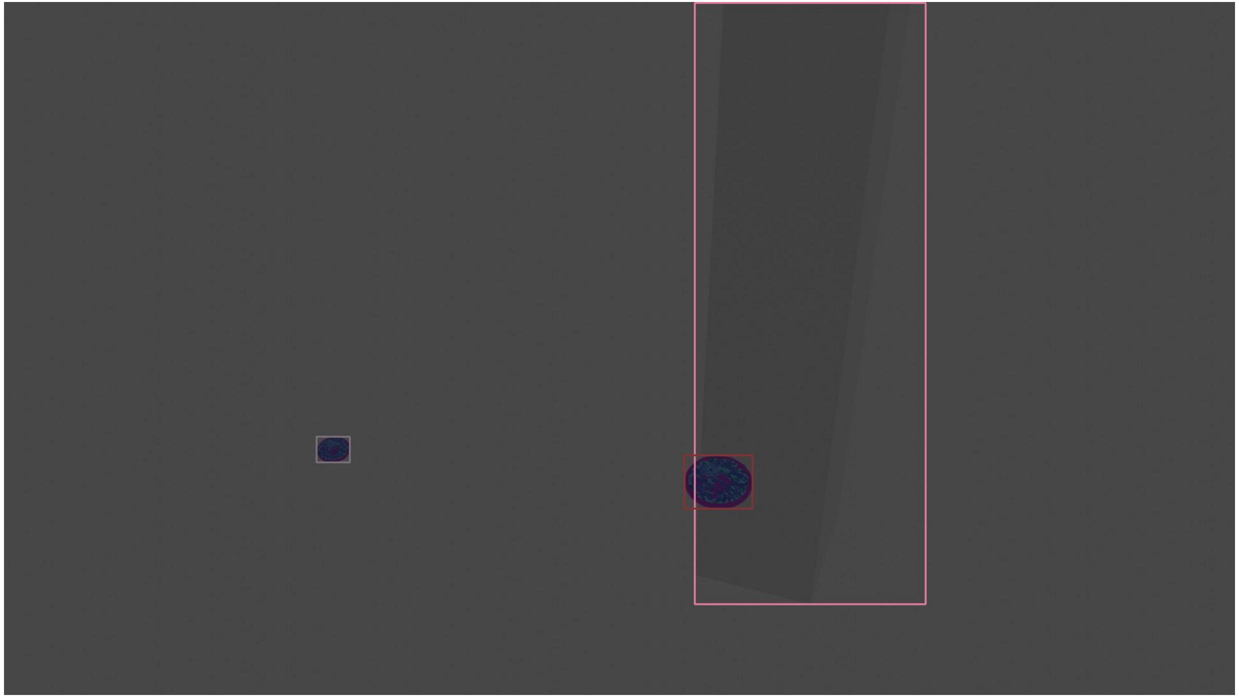


Foreign Object Occlusion



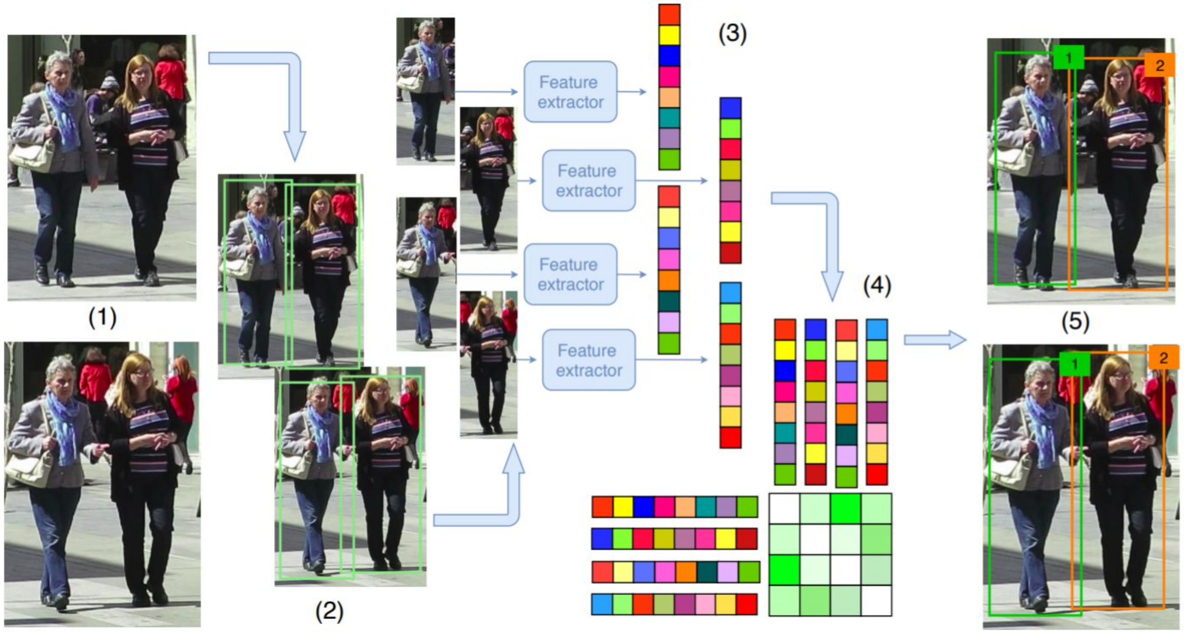


Foreign Object Occlusion



Workflow of multi-obejct tracking algorithm with appearance information

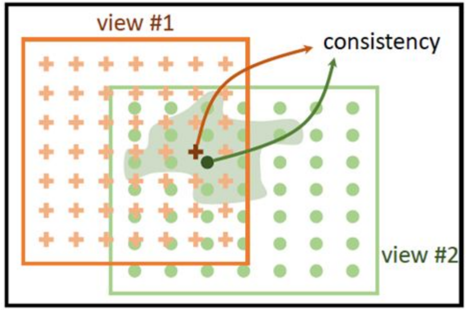
- A detector runs to obtain the bounding boxes of the objects
- For every bounding boxes, visual features are computed by a feature extractor
- Compute the similarity or distance between features of bboxs
- An association step matches corresponding bboxs in two frames and assigns a numerical id to each track

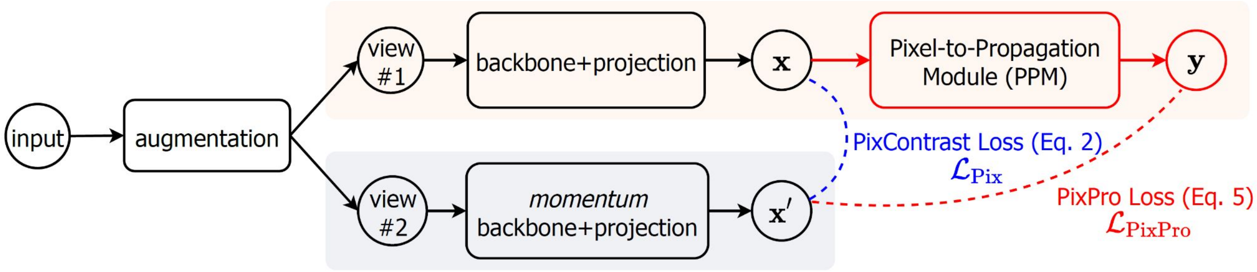


Source: Deep Learning in Video Multi-Object Tracking: A Survey, Ciapparone et. al.

PixPro

- PixPro outputs visual features using self-supervised learning.





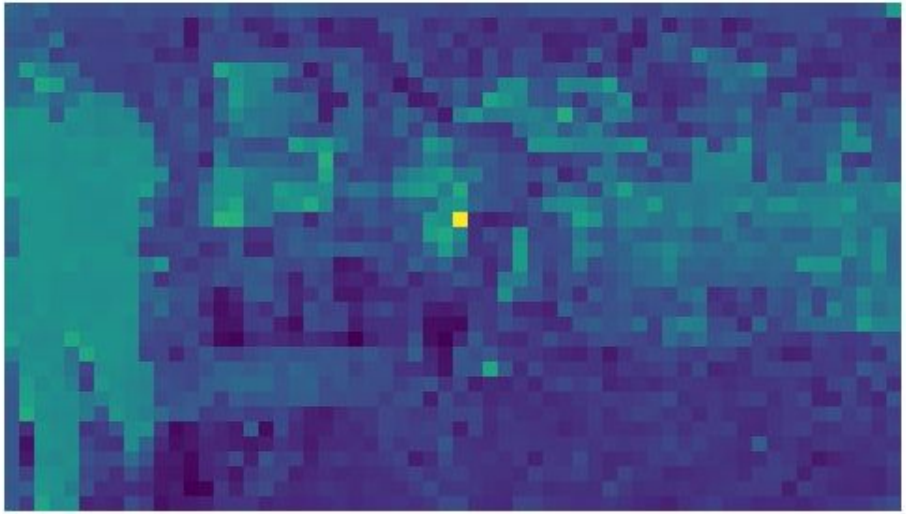
Source: Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning, Xie, Zhenda, et al.

- There is already a pre-trained PixPro model on the general dataset ImageNet. It’s not trained on a specific target dataset. This model can output visual features with small cosine distances for similar pixels. We can use the PixPro model to be the feature extractor in the workflow.

Visualization of PixPro model in pixel level

- The output of PixPro model is a feature map with 256 channels.
- Take the feature vector of the center point of a person as an example and compute the dot product between it and all the pixels in the feature map.
- The result shows the pretrained PixPro model can extract appearance infomation of objects.

Original image



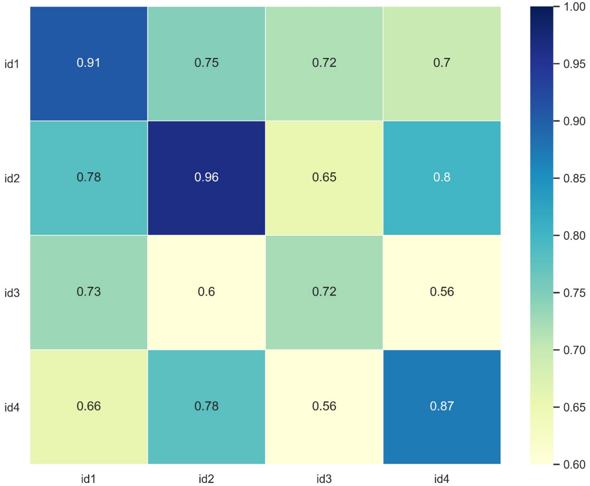
Visualization of feature map

Visualization of PixPro model in bbox level

- Take the mean values of feature vectors in bounding boxes and compute the dot product between all bboxs in two frames.
- This matrix is then used in Hungarian algorithm to match bboxs in two frames.

Frame 5

Frame 1



Overview of default and modified models

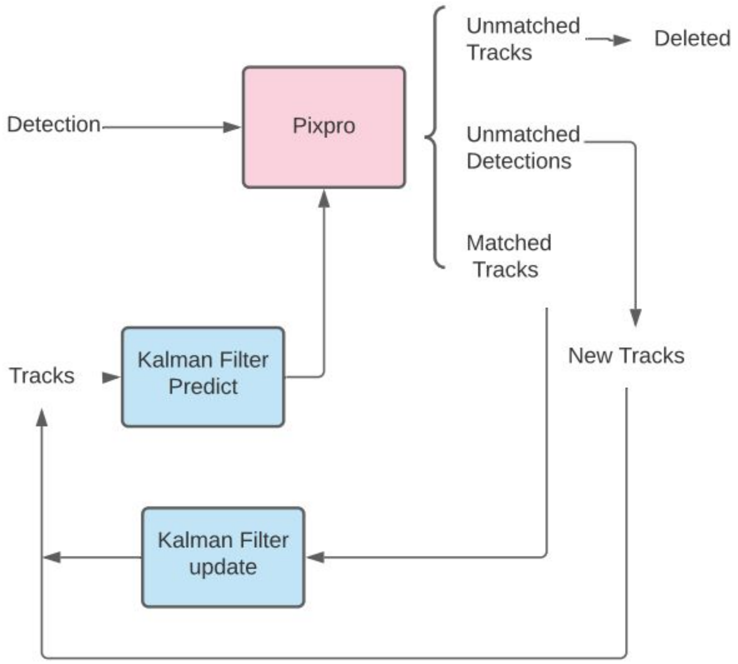
Properties Methods	Matching metric	Features cached	Model training
Default 1 SORT	IOU	No	No
Default 2 DeepSORT	IOU Visual features	Yes	Pretrained required for each customer dataset
Modified 1 PixProSORT	Visual features	No	PixPro algorithm pretrained for general dataset
Modified 2 PixProSORT with cached features	Visual features	Yes	
Modified 3 DeepSORT + PixPro	IOU Visual features	Yes	

Modified1: PixProSORT

Properties / Methods	Matching metric	Features cached	Model training
Default 1 SORT	IOU	No	No
Default 2 DeepSORT	IOU Visual features	Yes	Pretrained required for each customer dataset
Modified 1 PixProSORT	Visual features	No	PixPro algorithm pretrained for general dataset
Modified 2 PixProSORT with cached features	Visual features	Yes	
Modified 3 DeepSORT + PixPro	IOU Visual features	Yes	

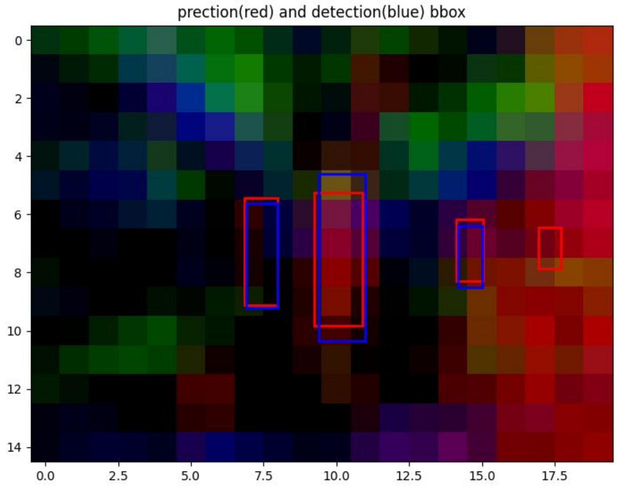
Modified 1: PixProSORT

- IOU is replaced by PixPro as the association criterion.
- PixPro extracts visual features of each frame.
- Given Bounding box coordinates, feature vector of each object can be computed.
- Cosine distance is used for assignment .



Modified 1: PixProSORT

- The example shows how the matching matrix is computed for a frame.
- Methods for computing pixel value of a bounding box has impact on the performance of model.



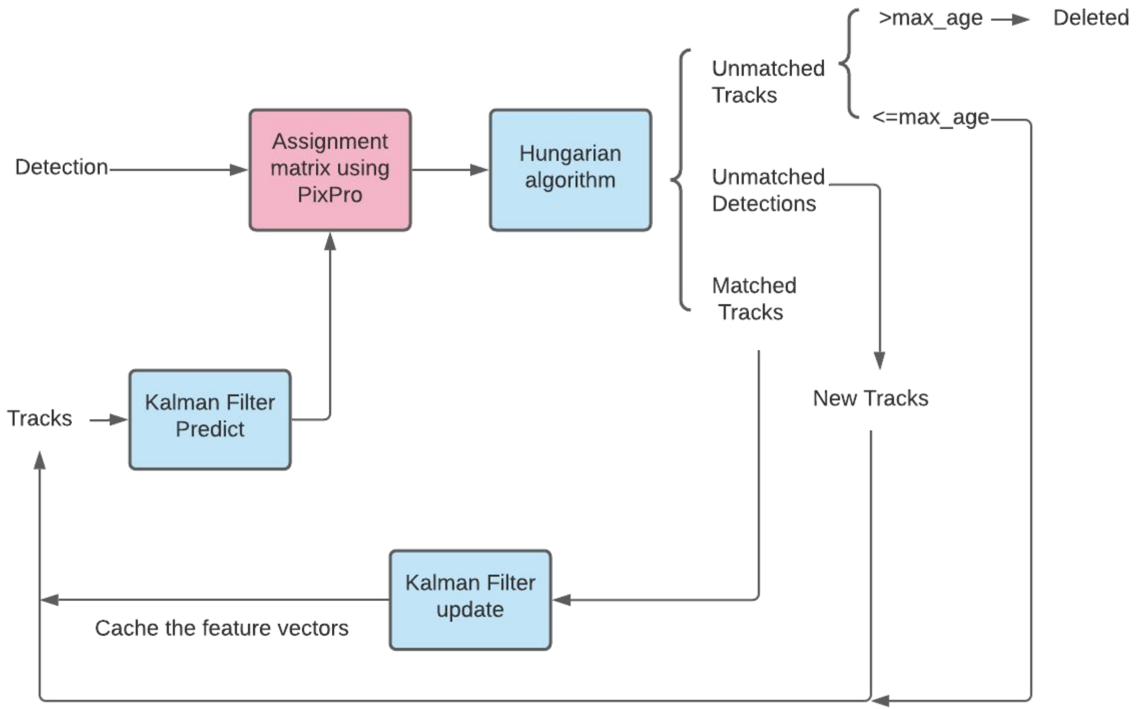
In one frame: 3 detections + 4 tracks		
Method	IOU (Intersection over union)	Pixpro
Representation matrix	detections: 3x4 tracks: 4x4	detections: 3x256 tracks: 4x256
Hungarian algorithm	Assignment matrix: 3x4	Assignment matrix: 3x4

Modified 2: PixProSORT with cached features

Methods \ Properties	Matching metric	Features cached	Model training
Default 1 SORT	IOU	No	No
Default 2 DeepSORT	IOU Visual features	Yes	Pretrained required for each customer dataset
Modified 1 PixProSORT	Visual features	No	PixPro algorithm pretrained for general dataset
Modified 2 PixProSORT with cached features	Visual features	**Yes**	
Modified 3 DeepSORT + PixPro	IOU Visual features	Yes	

Modified 2: PixProSORT with cached features

- Unmatched tracks won’t be deleted immediately.
- At Kalman Filter update step, the feature vectors are cached in memory.
- Use all the cached features and the current feature afte KF prediction to compute the assignment matrix.
- It helps to deal with occlusion.



Modified 2: PixProSORT with cached features

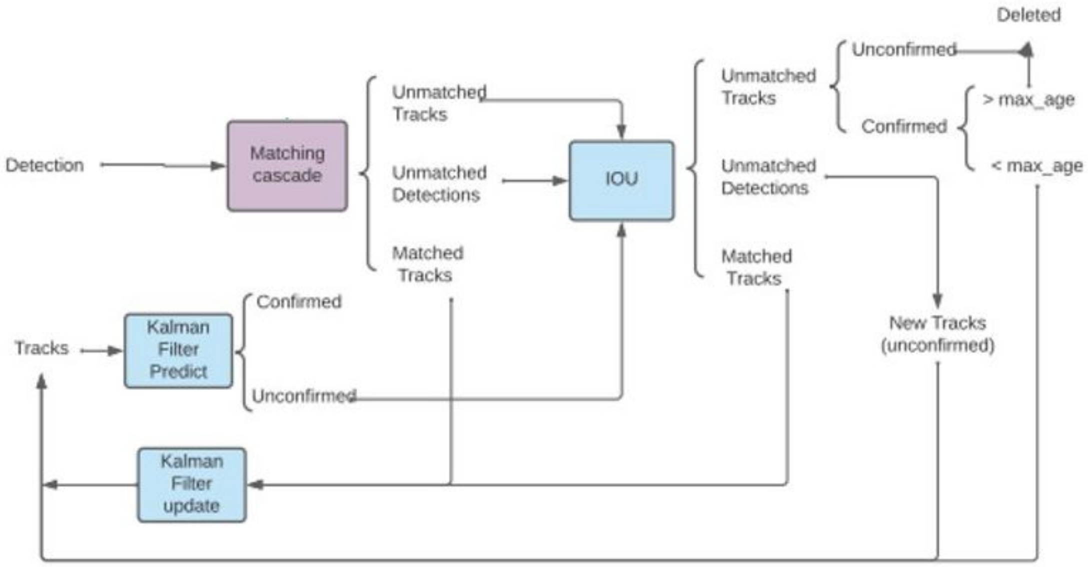
The person with id3 is occluded by the person with id2 in the middle frame. When the person with id3 is observable in the scene again, its id is not switched to another id. This means PixproSORT with cached features can deal with occlusion.

Modified 3: DeepSORT + PixPro

Methods \ Properties	Matching metric	Features cached	Model training
Default 1 SORT	IOU	No	No
Default 2 DeepSORT	IOU Visual features	Yes	Pretrained required for each customer dataset
Modified 1 PixProSORT	Visual features	No	PixPro algorithm pretrained for general dataset
Modified 2 PixProSORT with cached features	Visual features	Yes	
Modified 3 DeepSORT + PixPro	IOU Visual features	Yes	

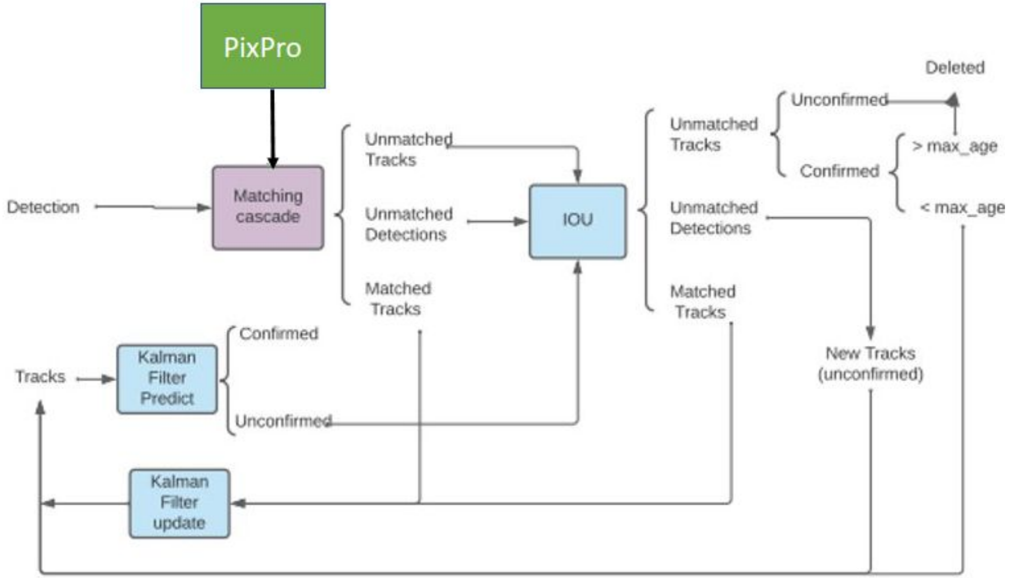
Modified 3: DeepSORT + PixPro

- DeepSORT
  - Integrating appearance information.
  - Two core algorithm: Matching cascade and IOU.
  - For each customer datasets the deep association metric feature representation must be extracted and stored before employing the algorithm.



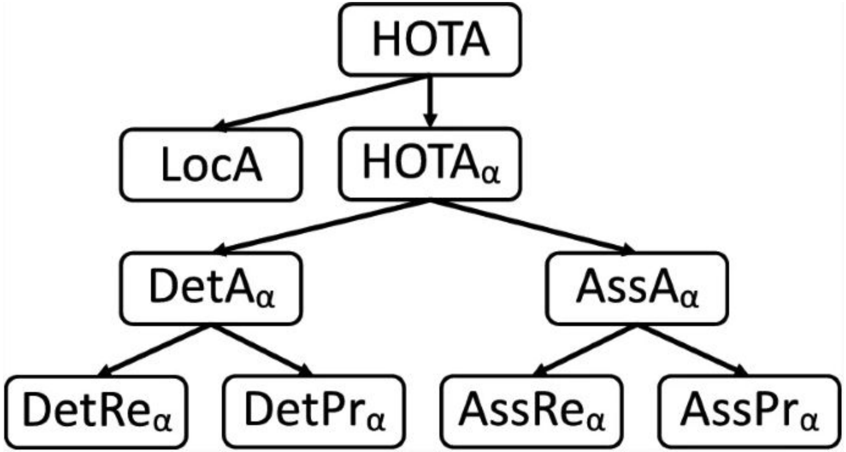
Modified 3: DeepSORT + PixPro

- The PixPro algorithm is integrated in Cascade algorithm.
- Feature representation is first computed and stored during the process.
- No offline pretrain of deep association metrics required.
- Time consumed

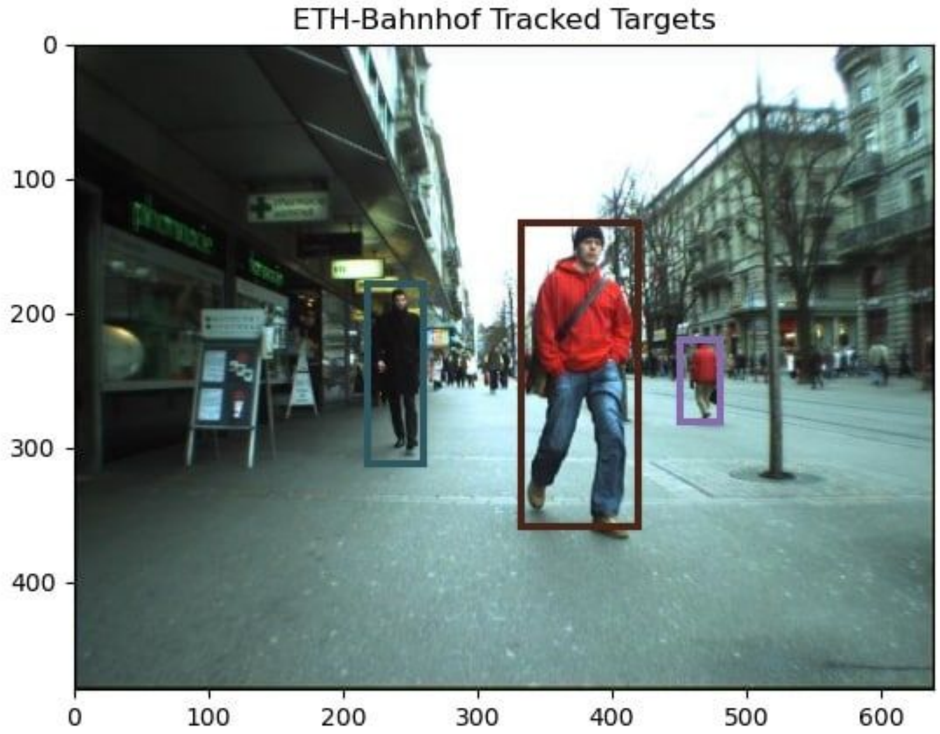


Results - Used Metrics

- **α**: Localization
- **Detection Recall:** TP / (TP + FN)
- **Detection Precision:** TP / (TP + FP)
- **Association Recall:** Errors occur if 2 different IDs are assigned to the same object
- **Association Precision:** Errors occur if a single ID is assigned to different objects

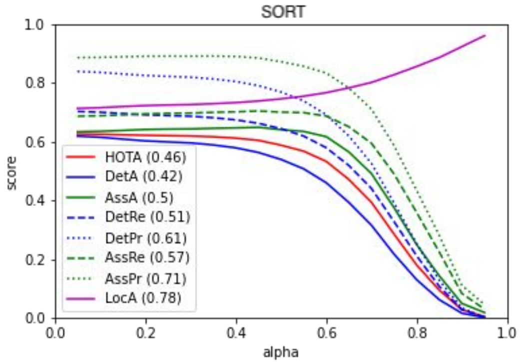


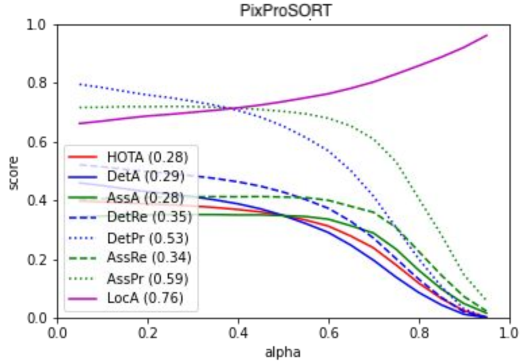
Results - Scene from MOT15



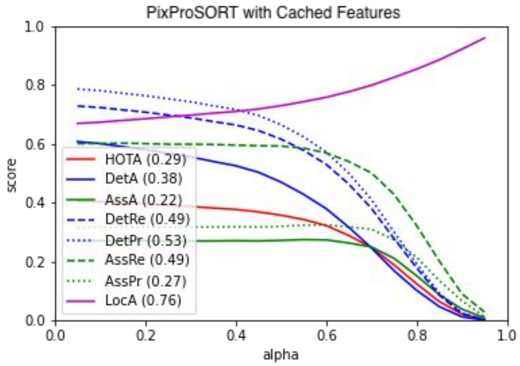
Results - Scene from MOT15

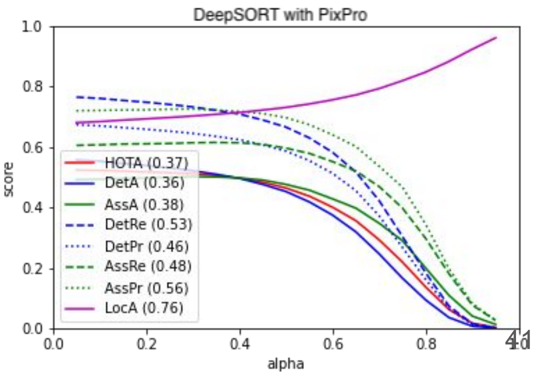
- SORT has the best performance





Tracker	SORT	DeepSORT with PixPro	PixProSORT with Cached Features	PixPro SORT
HOTA Score	**0.46**	**0.37**	**0.29**	**0.28**

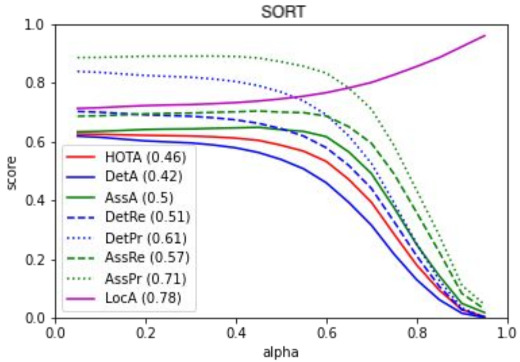


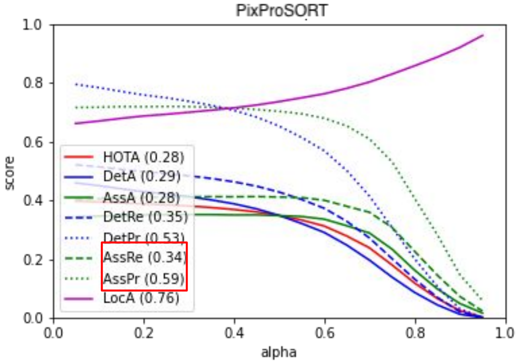


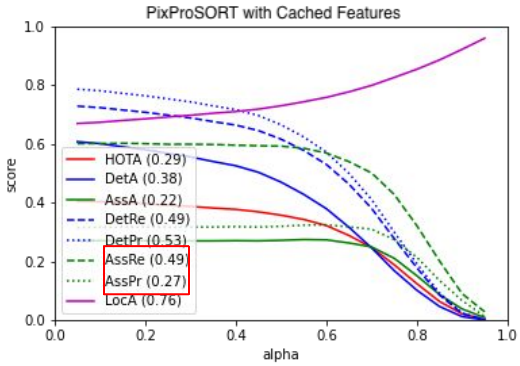
Results - Scene from MOT15

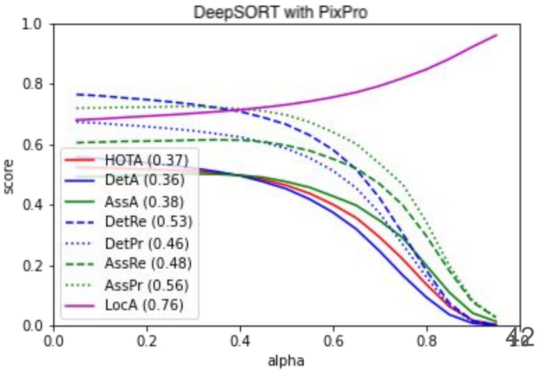
- SORT has the best performance
- Caching features increased recall while decreasing precision

	PixPro SORT	PixProSORT with Cached Features
Ass. Recall	**0.34**	**0.49**
Ass. Precision	**0.59**	**0.27**

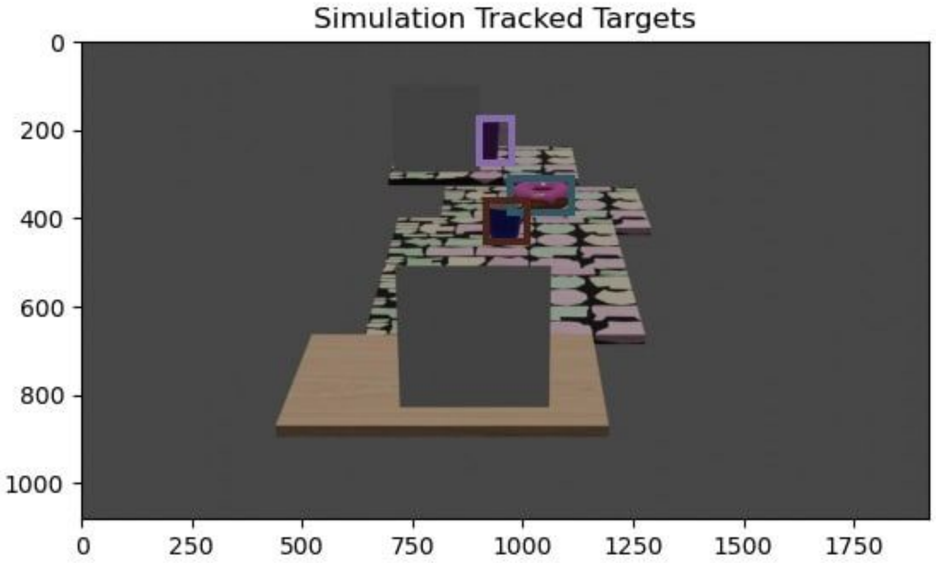






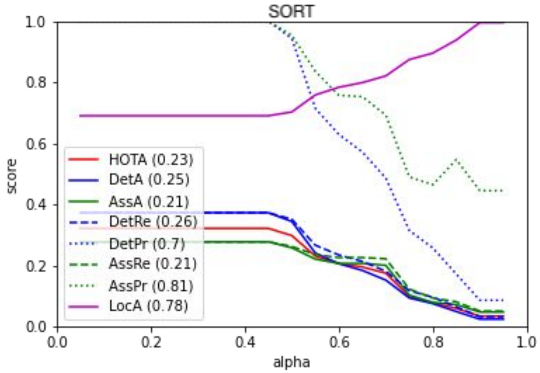


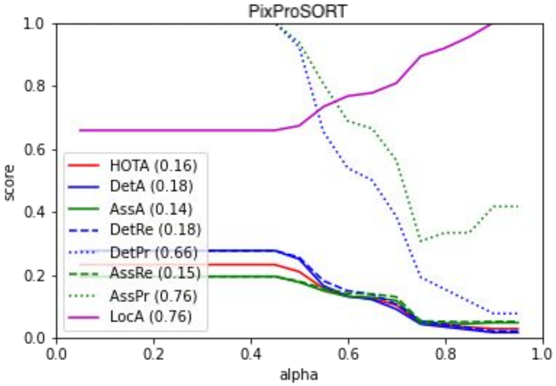
Results - Simulated Scene



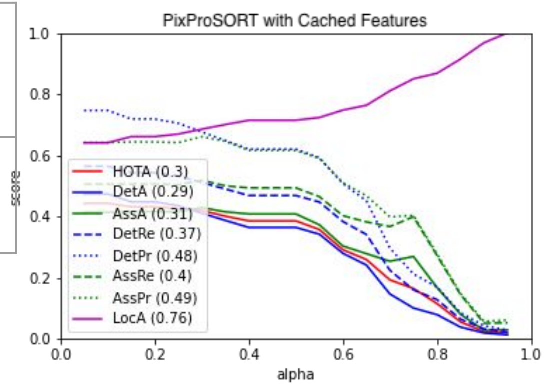
Results - Simulated Scene

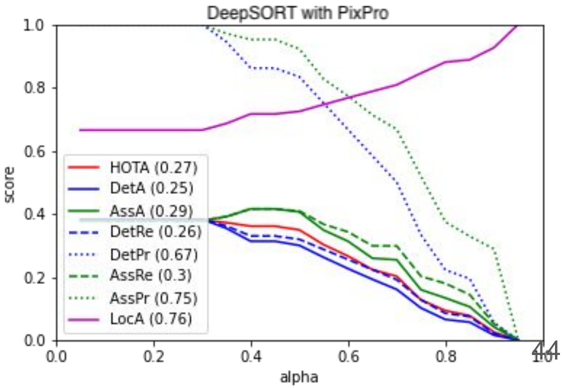
- PixProSORT with Cached Features has the best performance
  - High occlusion increases the importance of recall





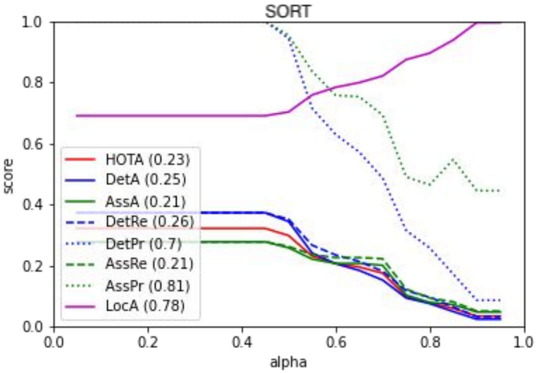
Tracker	PixProSORT with Cached Features	DeepSORT with PixPro	SORT	PixPro SORT
HOTA Score	**0.3**	**0.27**	**0.23**	**0.16**

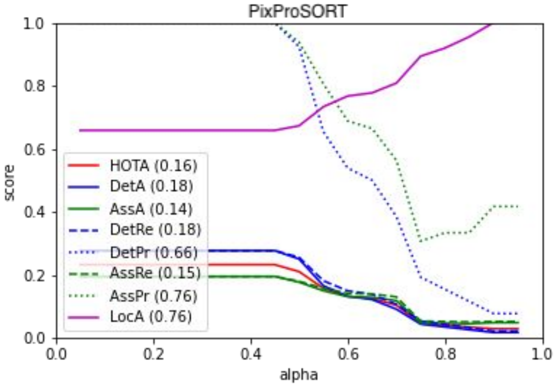




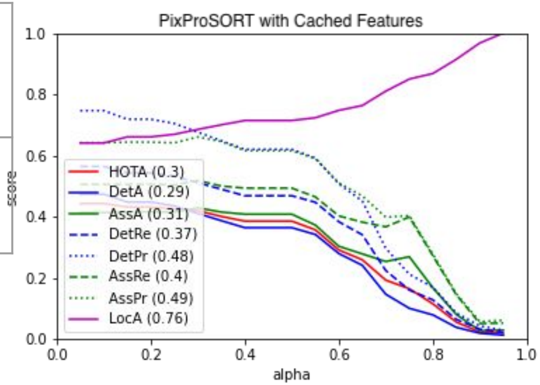
Results - Simulated Scene

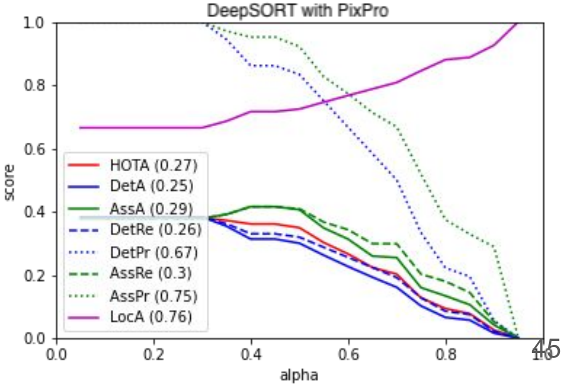
- PixProSORT with Cached Features has the best performance
- DeepSORT has the second best performance again





Tracker	PixProSORT with Cached Features	DeepSORT with PixPro	SORT	PixPro SORT
HOTA Score	**0.3**	**0.27**	**0.23**	**0.16**





Conclusion

- Contribution
  - Analysis of SORT
  - Introducing PixPro as the backbone of 3 new algorithms
  - Scene simulation for modes of failure
  - Evaluation and analysis of novelty algorithms
- Future work
  - Finetune PixPro with dataset suitable for foodtracking
  - Simulate more data to evaluate modes of failure

Thank you for your attention!