3D MOT with DPE (2022)

- Indoor
- Object detection + reconstruction + recognition
- MOTA Score
- New Dataset (MOTFRONT)

3D MOT with DPE (2022)

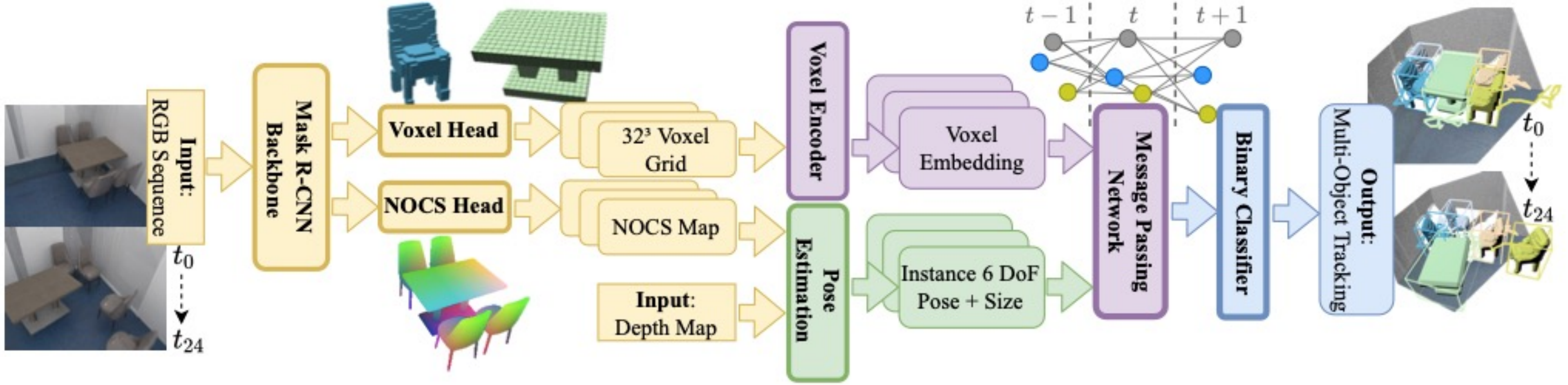


Figure 2: Network architecture overview: a Mask R-CNN backbone detects object instances from an RGB sequence, additionally predicting a 32^3 voxel grid and a NOCs map for each object which is aligned by a pose estimation module using supplementary depth sequence data represented as a depth point cloud, outputting a 7-DoF pose. A 3D convolutional network encodes predicted voxel grids into initial node embeddings. For a consistent object tracking we employ a graph neural network with a consecutive binary classifier, predicting active and non-active graph connections, which is constructed from geometry and pose embeddings and jointly optimizes reconstruction, pose, and object association.

3D MOT with DPE (2022)

👍

- Graph approach for differential pose estimation
- Joint reconstruction & pose estimation to achieve robustness
- Algorithm uses 5 frames at time

3D MOT with DPE (2022)

⛔

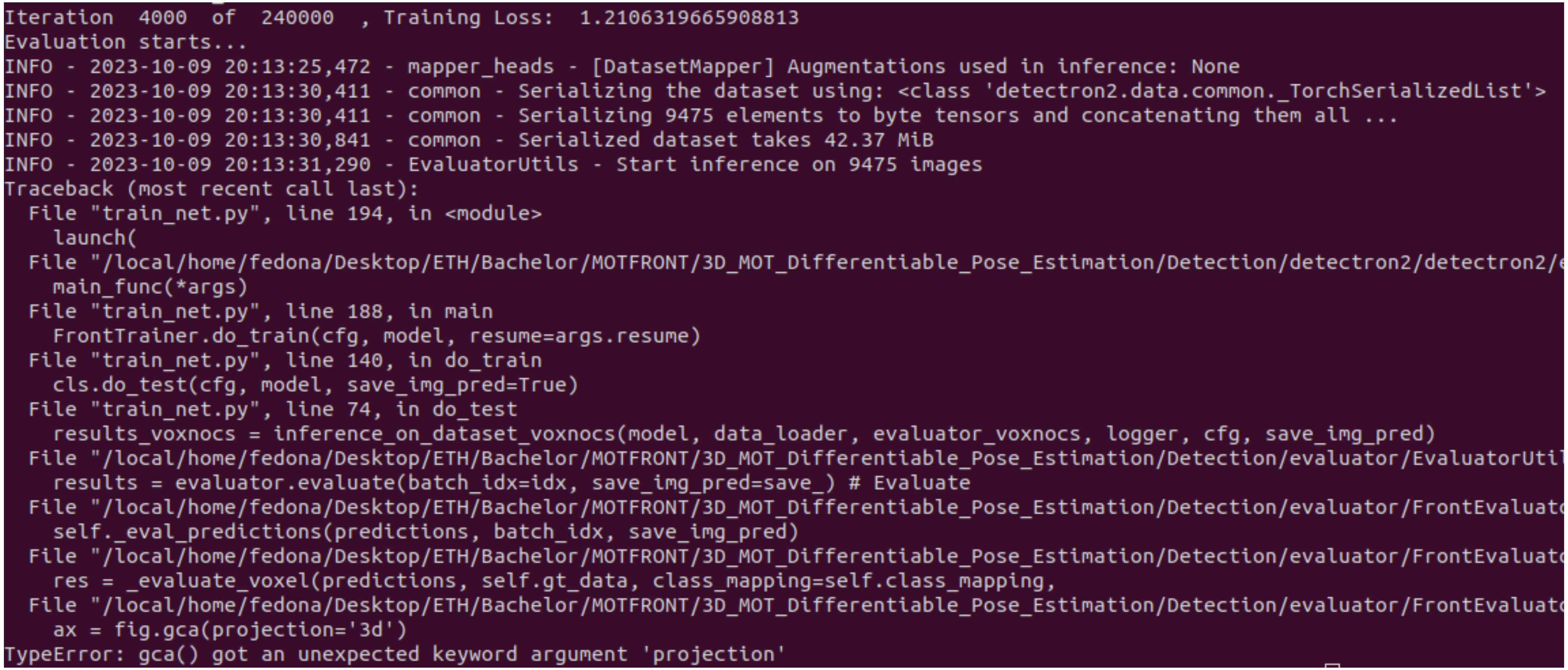
- Compared with “State-of-the-Art” Seeing Behind Objects (2020)
  - SBO is the precedent paper from the same research team
  - MOTA results do not match, they use the new dataset and get 4.5% difference compared when using DYNSYNTH
  - SBO uses 2 frames, unfair?
- Looks poorly tested, should have also used more an old dataset
- Basically optimized version of SBO that has good performance on the new dataset

3D MOT with DPE (2022)

⛔

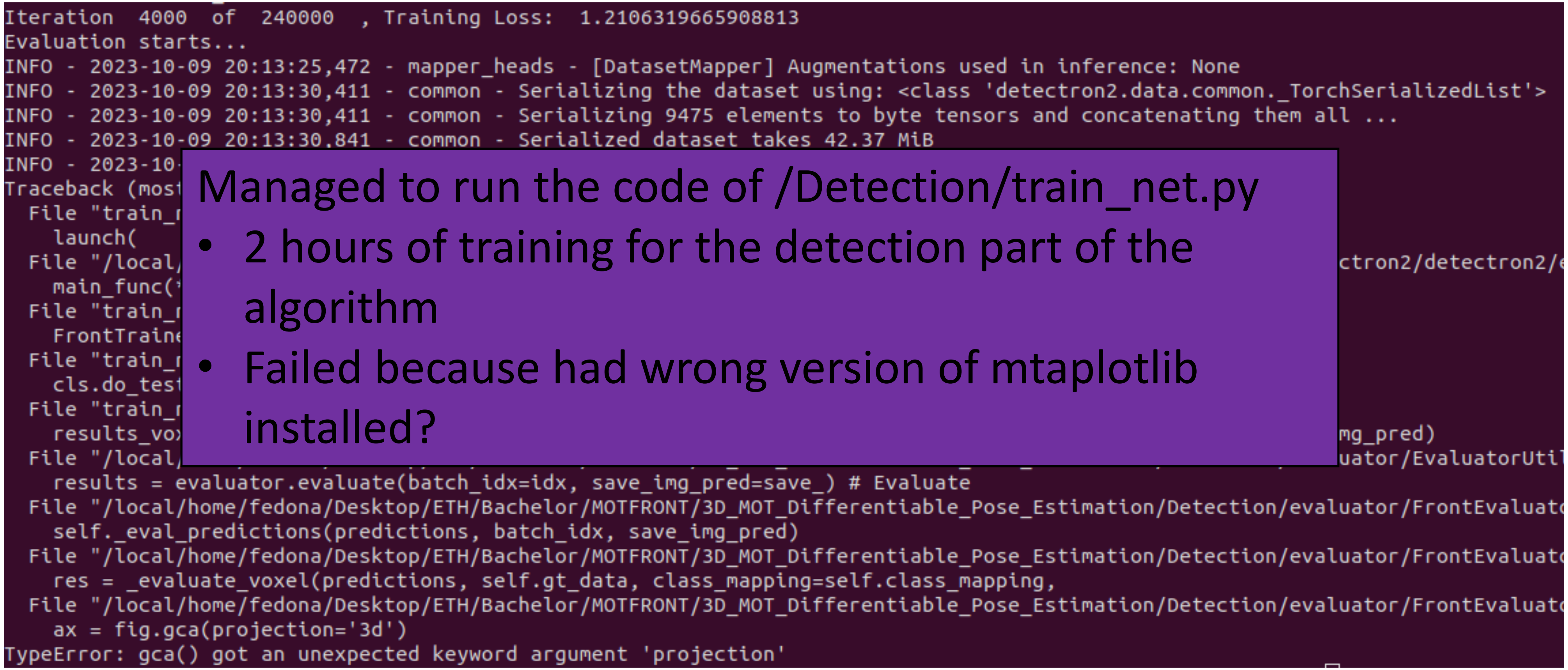
- Compared with “State-of-the-Art” Seeing Behind Objects (2020)
  - SBO is the precedent paper from the same research team
  - MOTA results do not match, they use the new dataset and get 4.5% difference compared when using DYNSYNTH
  - SBO uses 2 frames, unfair?
- Looks poorly tested, should have also used more an old dataset
- Basically optimized version of SBO that has good performance on the new dataset
- Hard to set up the environment, large number of dependencies, ”pip install vision3d” can not be found
- Had to manually insert dataset and change folder’s names to make it run
- 0 issues in the github page of the project

3D MOT with DPE (2022)



```
Iteration  4000  of  240000  , Training Loss:  1.2106319665908813
Evaluation starts...
INFO - 2023-10-09 20:13:25,472 - mapper_heads - [DatasetMapper] Augmentations used in inference: None
INFO - 2023-10-09 20:13:30,411 - common - Serializing the dataset using: <class 'detectron2.data.common._TorchSerializedList'>
INFO - 2023-10-09 20:13:30,411 - common - Serializing 9475 elements to byte tensors and concatenating them all ...
INFO - 2023-10-09 20:13:30,841 - common - Serialized dataset takes 42.37 MiB
INFO - 2023-10-09 20:13:31,290 - EvaluatorUtils - Start inference on 9475 images
Traceback (most recent call last):
  File "train_net.py", line 194, in <module>
    launch(
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/detectron2/detectron2/e
    main_func(*args)
  File "train_net.py", line 188, in main
    FrontTrainer.do_train(cfg, model, resume=args.resume)
  File "train_net.py", line 140, in do_train
    cls.do_test(cfg, model, save_img_pred=True)
  File "train_net.py", line 74, in do_test
    results_voxnocs = inference_on_dataset_voxnocs(model, data_loader, evaluator_voxnocs, logger, cfg, save_img_pred)
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/evaluator/EvaluatorUtil
    results = evaluator.evaluate(batch_idx=idx, save_img_pred=save_) # Evaluate
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/evaluator/FrontEvaluato
    self._eval_predictions(predictions, batch_idx, save_img_pred)
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/evaluator/FrontEvaluato
    res = _evaluate_voxel(predictions, self.gt_data, class_mapping=self.class_mapping,
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/evaluator/FrontEvaluato
    ax = fig.gca(projection='3d')
TypeError: gca() got an unexpected keyword argument 'projection'
```

3D MOT with DPE (2022)



```
Iteration  4000  of  240000  , Training Loss:  1.2106319665908813
Evaluation starts...
INFO - 2023-10-09 20:13:25,472 - mapper_heads - [DatasetMapper] Augmentations used in inference: None
INFO - 2023-10-09 20:13:30,411 - common - Serializing the dataset using: <class 'detectron2.data.common._TorchSerializedList'>
INFO - 2023-10-09 20:13:30,411 - common - Serializing 9475 elements to byte tensors and concatenating them all ...
INFO - 2023-10-09 20:13:30,841 - common - Serialized dataset takes 42.37 MiB
INFO - 2023-10-
Traceback (most
  File "train_
    launch(
  File "/local,                                                                                              ctron2/detectron2/e
    main_func(
  File "train_
    FrontTraine
  File "train_
    cls.do_test
  File "train_
    results_vox                                                                                              mg_pred)
  File "/local,                                                                                              uator/EvaluatorUtil
    results = evaluator.evaluate(batch_idx=idx, save_img_pred=save_) # Evaluate
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/evaluator/FrontEvaluato
    self._eval_predictions(predictions, batch_idx, save_img_pred)
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/evaluator/FrontEvaluato
    res = _evaluate_voxel(predictions, self.gt_data, class_mapping=self.class_mapping,
  File "/local/home/fedona/Desktop/ETH/Bachelor/MOTFRONT/3D_MOT_Differentiable_Pose_Estimation/Detection/evaluator/FrontEvaluato
    ax = fig.gca(projection='3d')
TypeError: gca() got an unexpected keyword argument 'projection'
```

Managed to run the code of /Detection/train_net.py

- 2 hours of training for the detection part of the algorithm
- Failed because had wrong version of mtaplotlib installed?

OBJECT FUSION (2022)

- Goal: introduce an efficient and performing object reconstruction method
- Focus on efficiency
- Sliding keyframe window (5-10 frames)
- Pose estimation and reconstruction go get a 3D map of the scene
- SceneNet & ScanNet datasets

OBJECT FUSION (2022)

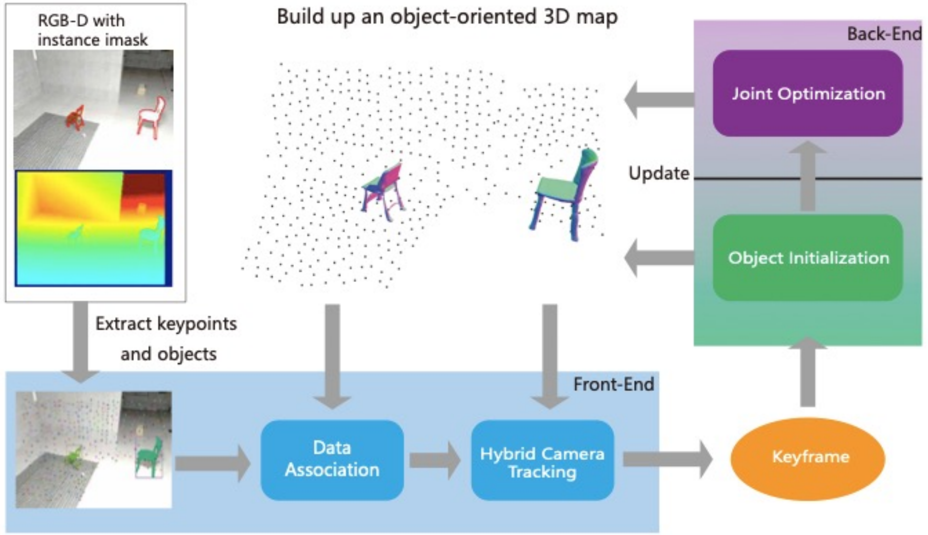


Figure 2. Overview of our ObjectFusion based on deep implicit object representation. ObjectFusion estimates the camera pose of each frame and incrementally builds up 3D surface reconstruction of object instances in the scene.

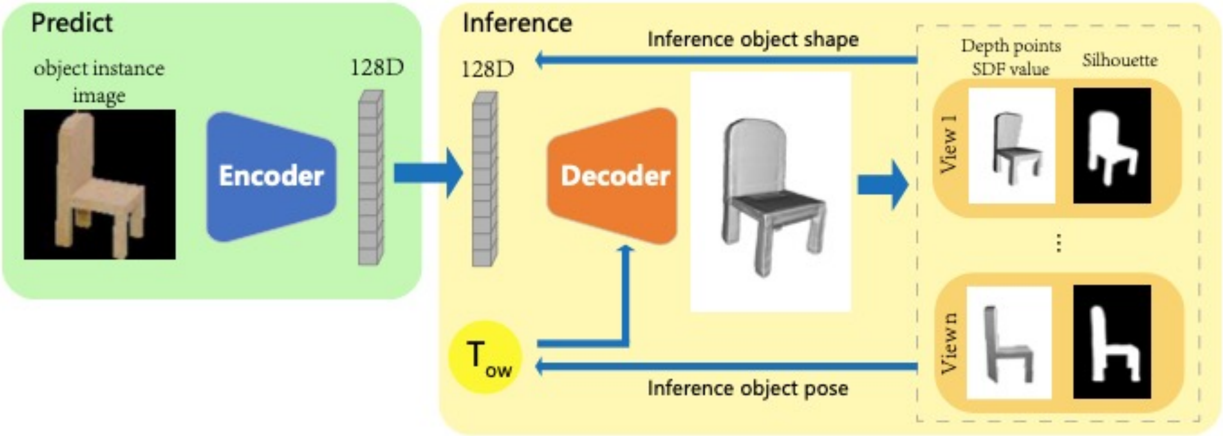


Figure 3. The backbone of our deep implicit object representation. The encoder encodes an object instance image as a latent vector, and then is decoded as a signed distance function of the object. The signed distance value of surface points (depth) and projection silhouette are used object shape and pose inference.

OBJECT FUSION (2022)

- Detect instance segmentation mask of frame
- Encode object to “latent vector”
- Iteratively update object latent vector & pose using hybrid cues
- Joint optimization of object shape, pose and camera pose on a sliding window of frames (5 to 10 frames)

OBJECT FUSION (2022)

👍

- New object representation method, encoded through simple Encoding/Decoding Network
- Great reconstruction quality

OBJECT FUSION (2022)

⛔

- Evaluation strategy is not fine grained, it becomes hard to understand the actual contribution of each optimization
- Unclear what they mean with ”Ours (w/o Obj)”
  - “No object landmarks to evaluate the effect of object term”??
- Still worst results than ORB (2015) in pretty much each scene

Seeing Behind Objects (2020)

- Indoor
- Focus on complete object geometry
- 72h training
- DYNSYNTH & SCANNET datasets

Seeing Behind Objects (2020)

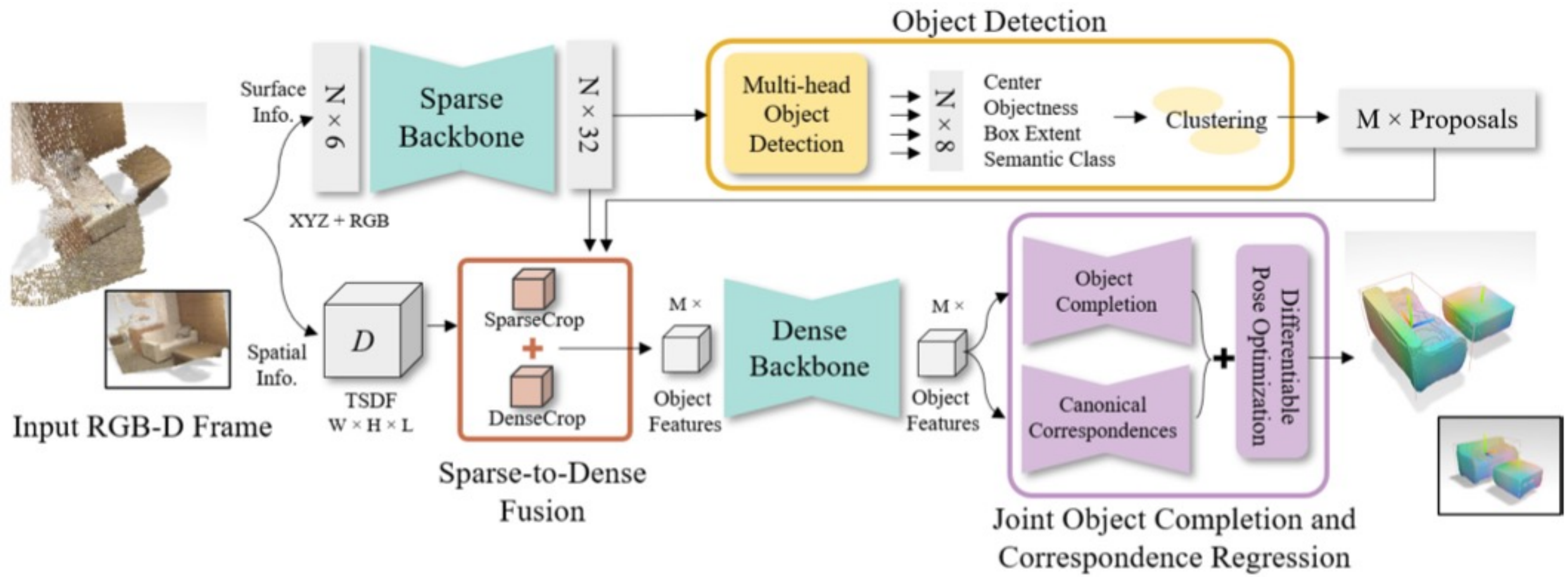


Figure 2. Overview of our network architecture for joint object completion and tracking. From a TSDF representation of an RGB-D frame, we employ a backbone of sparse 3D convolutions to extract features. We then detect objects characterized by 3D bounding boxes, and predict for each object both the complete object geometry beyond the view observation as well as dense correspondences a canonical space; the correspondences on the complete geometry then inform a differentiable pose optimization to produce object pose estimates and within-frame dense correspondences. By predicting correspondences not only in observed regions but also unobserved areas, we can provide strong correspondence overlap under strong object or camera motion, enabling robust dynamic object tracking.

Seeing Behind Objects (2020)

- Object detection via predicting objects boundary boxes
  - Encoder-decoder to extract features
- Sparse to dense fusion to get final object features
- Object completion
  - 3D Convolutions via encoder-decoder -> get dense features
- Object correspondences
- Object tracking based on previous results
  - Hungarian algorithm to find optimal assignment

Seeing Behind Objects (2020)

⛔

- Compared with DETECTRON Mask R-CNN, but this was made for object segmentation and not optimized for object tracking!
  - MASK R-CNN used on Resnet 101 dataset of real world outdoor images, it might overperform with indoor images

DYNA-SLAM (2018)

- Outdoor
- Addresses the issue of having dynamic objects in the scene
- Both monocular camera images and RGB-D
- 5 frames
- TUM and KITTI datasets
- 3D reconstruction

DYNA-SLAM (2018)

👍

- Achieves full dynamic object detection and localization
- Tracks the camera creating a static and reusable map of the scene
- Best in the TUM dataset

DYNA-SLAM (2018)

⛔

- Less accurate then ORB (2015) in KITTI dataset
  - But in scenes with many dynamic objects always performs better

BundleTrack (2021)

- Indoor
- Faces the challenge of NOT having CAD models of target objects!
- NOCS and YCBInEOAT datasets

BundleTrack (2021)

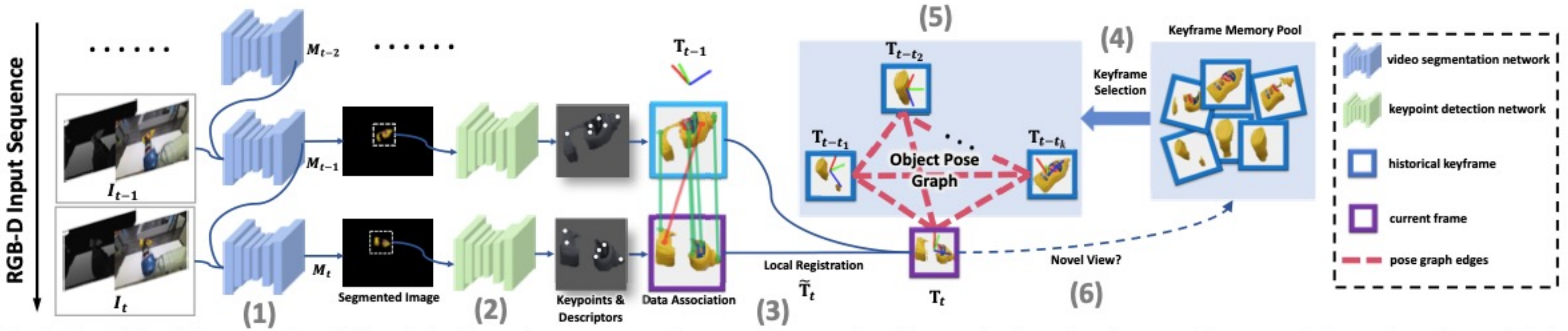


Fig. 2: *BundleTrack* framework from left to right: (1) an image segmentation network returns the object mask given the prior one; (2) a network detects keypoints and their descriptors; (3) keypoints are matched and coarse registration is performed between consecutive frames to estimate an initial relative transform $\tilde{\mathbf{T}}_t$; (4) keyframes are selected from a memory pool to participate in the pose graph optimization; (5) online pose graph optimization outputs a refined spatiotemporal consistent pose $\mathbf{T}_t$; and (6) the latest frame is included in the memory pool, if it is a novel view to enrich diversity.

BundleTrack (2021)

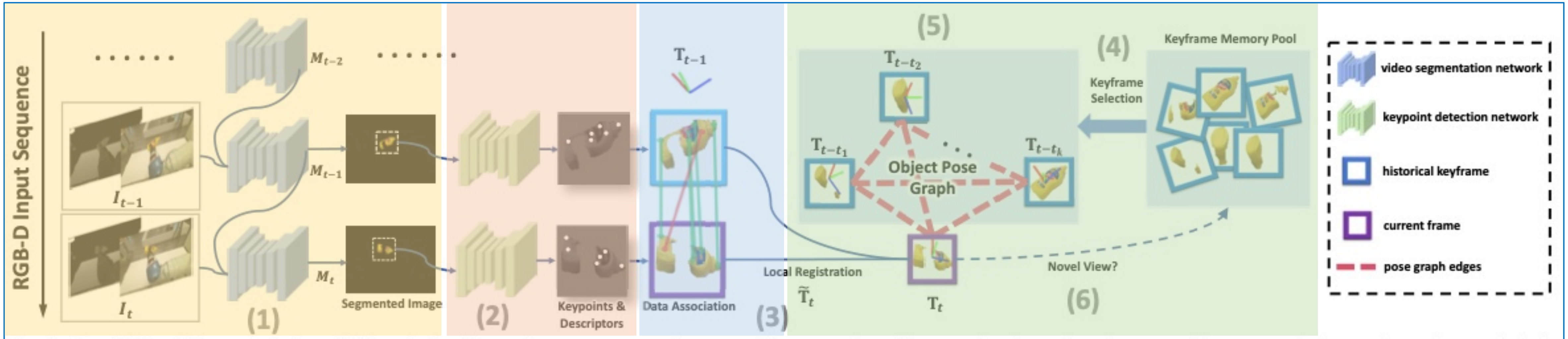


Fig. 2: *BundleTrack* framework from left to right: (1) an image segmentation network returns the object mask given the prior one; (2) a network detects keypoints and their descriptors; (3) keypoints are matched and coarse registration is performed between consecutive frames to estimate an initial relative transform $\tilde{\mathbf{T}}_t$; (4) keyframes are selected from a memory pool to participate in the pose graph optimization; (5) online pose graph optimization outputs a refined spatiotemporal consistent pose $\mathbf{T}_t$; and (6) the latest frame is included in the memory pool, if it is a novel view to enrich diversity.

1) Image segmentation
2) Keypoint detection
3) Data association to get initial coarse estimate
4) Object Pose Graph of meaningful Keyframes to compute an optimized pose for the current timestamp

BundleTrack (2021)

👍

- Achieves SOTA performance without any CAD model as prerequisite
  - No need to train over large number of CAD models
  - No more need of rigid dataset that contain a limited amount of object categories
- Modular pipeline
- Optimize for timing performance, framework runs at 10Hz avg.
  - In the keyframe selection use a greedy algorithm instead of using the “correct” algorithm

BundleTrack (2021)

⛔
- Does not perform object reconstruction
  - But has a useful Keyframe memory pool for that!
- Suffers from severe object occlusions
  - With better visibility in subsequent frames is able to recover
- Noisy segmentation
  - More advanced segmentation module could boost performance

BundleSDF (2023)

- Indoor
- Faces the challenge of NOT having CAD models of target objects!
- YCBInEOAT, HO3D and BEHAVE datasets
- Object reconstruction with texture

BundleSDF (2023)

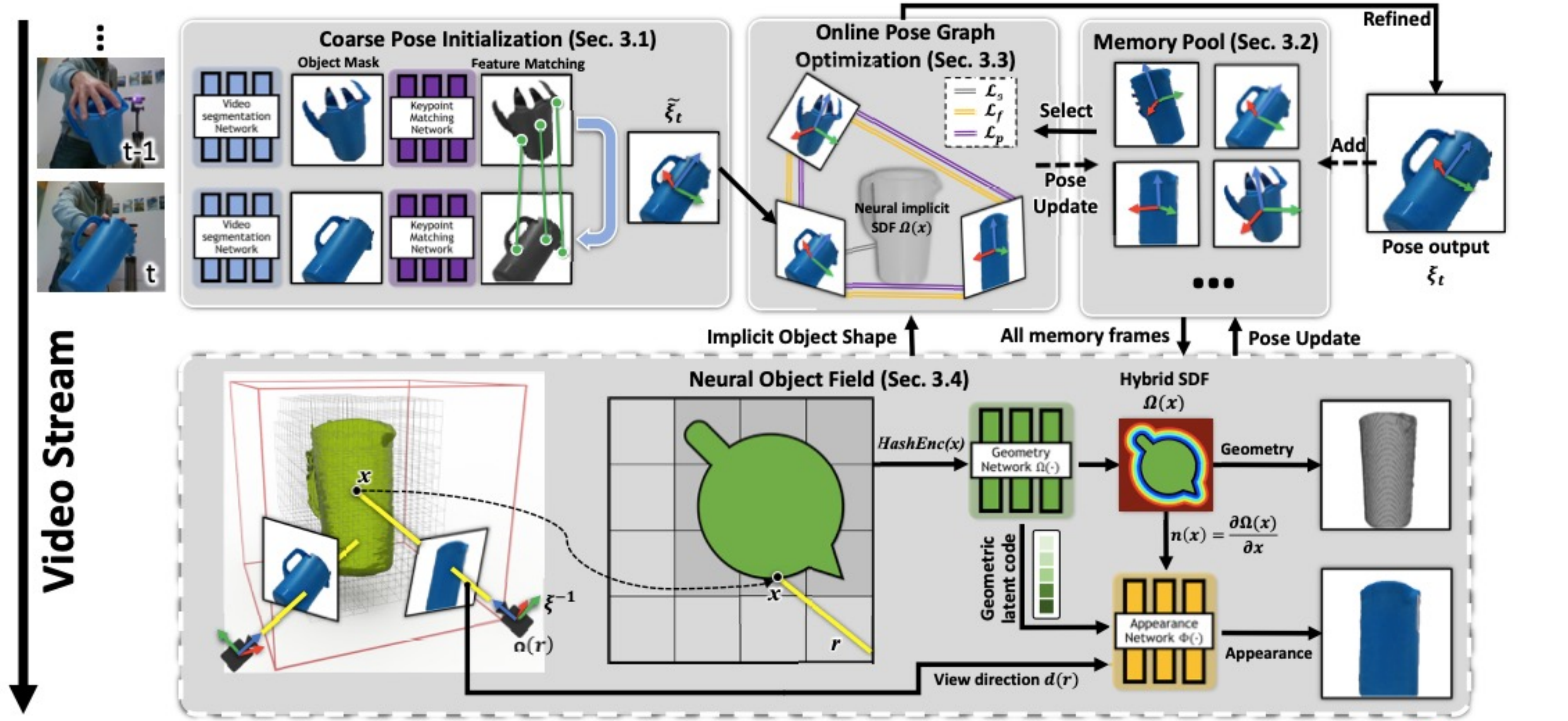


Figure 2. Framework overview. First, features are matched between consecutive segmented images, to obtain a coarse pose estimate (Sec. 3.1). Some of these posed frames are stored in a memory pool, to be used and refined later (Sec. 3.2). A pose graph is dynamically created from a subset of the memory pool (Sec. 3.3); online optimization refines all the poses in the graph jointly with the current pose. These updated poses are then stored back in the memory pool. Finally, all the posed frames in the memory pool are used to learn a Neural Object Field (in a separate thread) that models both geometry and visual texture (Sec. 3.4) of the object, while adjusting their previously estimated poses.

BundleSDF (2023)

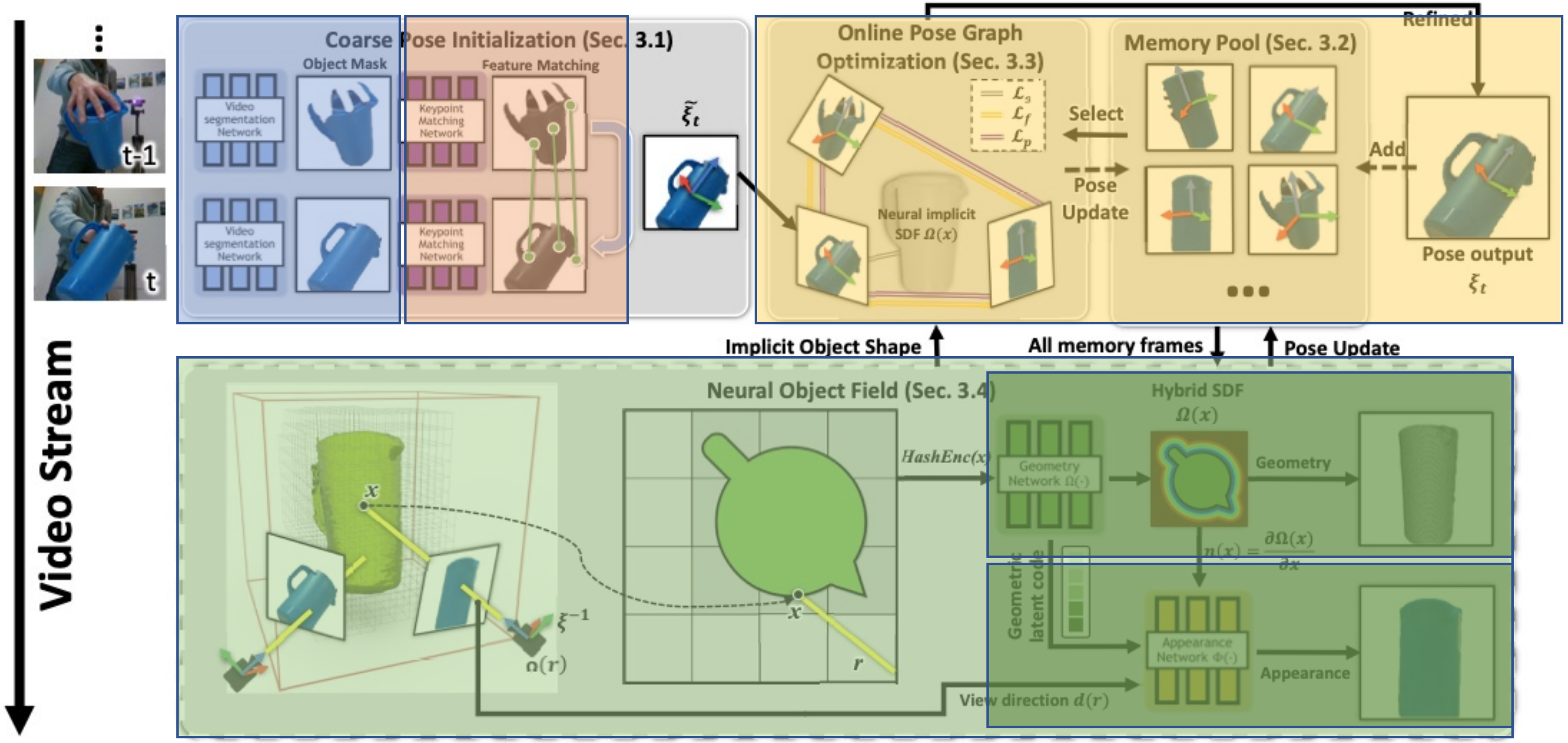
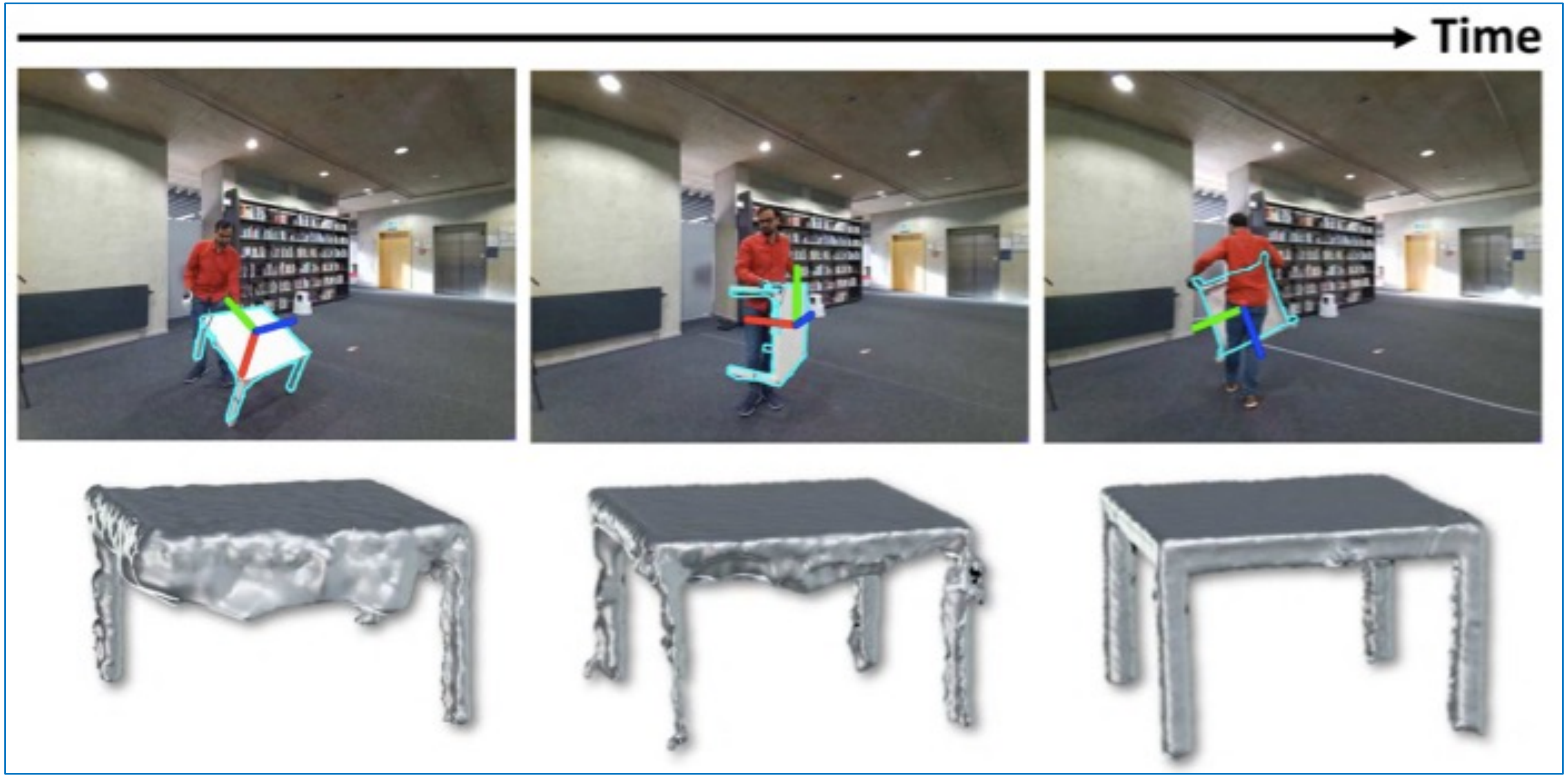


Figure 2. Framework overview. First, features are matched between consecutive segmented images, to obtain a coarse pose estimate (Sec. 3.1). Some of these posed frames are stored in a memory pool, to be used and refined later (Sec. 3.2). A pose graph is dynamically created from a subset of the memory pool (Sec. 3.3); online optimization refines all the poses in the graph jointly with the current pose. These updated poses are then stored back in the memory pool. Finally, all the posed frames in the memory pool are used to learn a Neural Object Field (in a separate thread) that models both geometry and visual texture (Sec. 3.4) of the object, while adjusting their previously estimated poses.

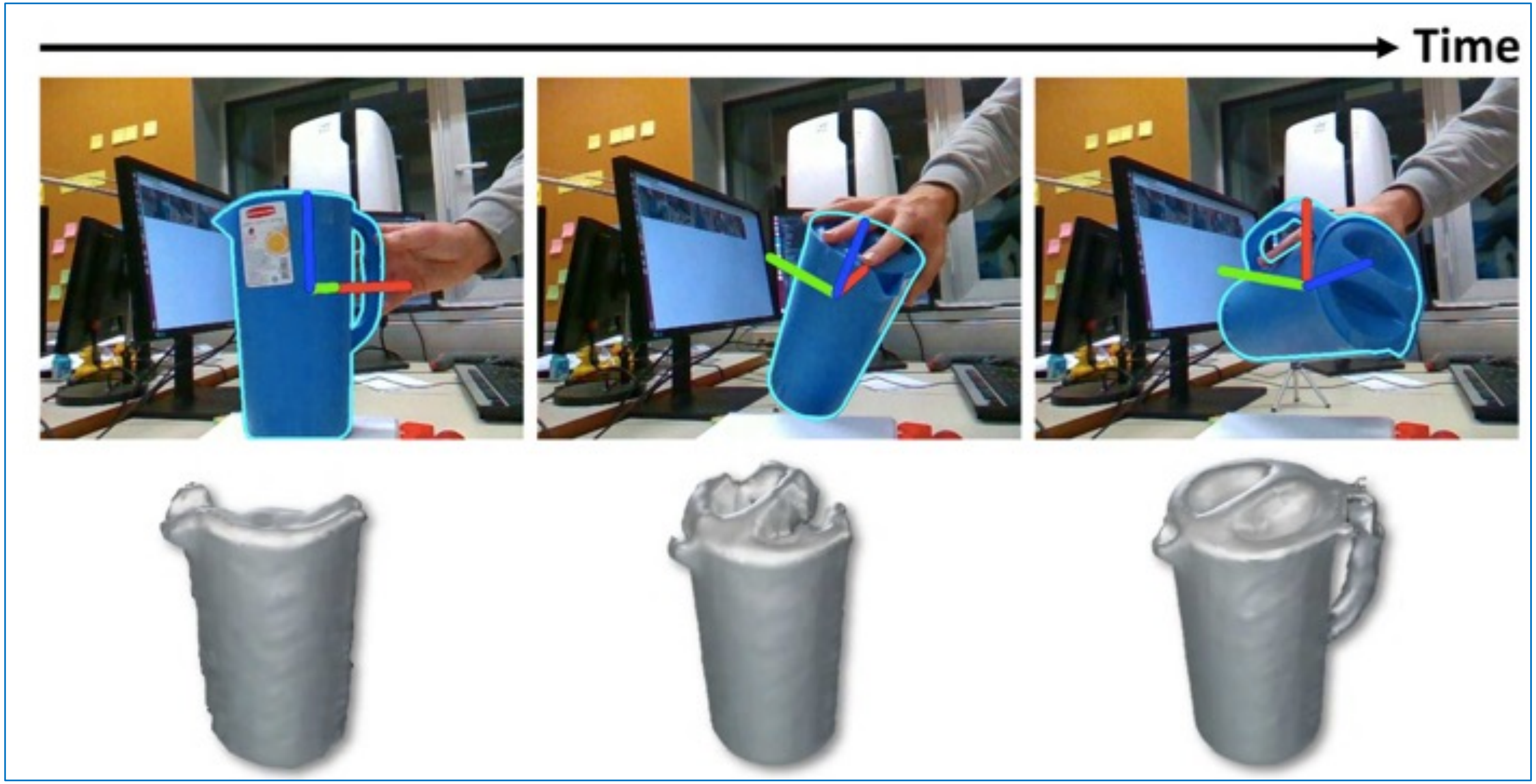
BundleSDF (2023)

- 👍 Achieves SOTA performance without any CAD model as prerequisite
  - No need to train over large number of CAD models
  - No more need of rigid dataset that contain a limited amount of object categories
- Modular pipeline
- Optimize for timing performance, framework runs at 10Hz avg.
- Great results with occluded objects!

BundleSDF (2023)



BundleSDF (2023)



BundleSDF (2023)

⛔

- Requires segmentation of the object in the initial frame
  - This was not the case for BundleTrack!
- Poor conclusion
  - Last sentence: “Future work will be aimed at leveraging shape priors to reconstruct unseen parts”
    - Destroys the goal of BundleTrack?

DATASETS

Name	Year	Content	Size	Other
MOTFront	2022	Computer generated realistic-scenes RGB-D images	2381 sequences; total 60k images	inspires from 3D-FRONT; MOTFRONT
KITTI	2012-2021	Real traffic scenarios, RGB, grayscale, 3D laser scanner	depends which category	Usually no ground truth for semantic segmentation; DYNASLAM
TUM-SLAM	2012	Real world indoor sequences; Kitnect RGB-D	circa 40 video sequences	DYNASLAM
ImageNet	2005-2017	Real world object, more than 20k categories	14 mio images	to train Resnet101
DYNSYNTH	?	RGB-D synthetic indoor scenes	3300 scenes	Can’t find it! SbO

DATASETS

Name	Year	Content	Size	Other
ScanNet	2017-2018	RGB-D dataset with 2D and 3D data; collection of voxels	2.5 mio views in more than 1500 scans; annotated	ObjectFusion, SbO
SceneNet	2016	RGB-D dataset of synthetic indoor scenes		ObjectFusion
NOCS	2019	RGB-D dataset of real indoor scenes + computer generated objects on real backgrounds	18 scenes, 6 object categories	BundleTrack
YCBInEOAT	2020	RGB-D real indoor dataset, with ground-truth poses; focus on occlusion and robot-manipulation	9 videos, 5 objects	BundleTrack