SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY - INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Robotics, Cognition, Intelligence

Improving Robustness of an Unknown Instance Segmentation Algorithm

Stella Dimitra Tragianni

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY - INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Robotics, Cognition, Intelligence

Improving Robustness of an Unknown Instance Segmentation Algorithm

Robustheitsverbesserung eines unbekannten Objektsegmentierungsansatzes

Author:	Stella Dimitra Tragianni
Supervisor:	PD Dr. habil. Rudolph Triebel
Advisor:	Wout Boerdijk, Maximilian Durner
Submission Date:	15 May 2023

I confirm that this master's thesis in robotics, cognition, intelligence is my own work and I have documented all sources and material used.

Munich, 15 May 2023 Stella Dimitra Tragianni

Acknowledgments

I would first like to warmly thank PD Dr. habil. Rudolph Triebel for the opportunity to conduct this master's thesis at the department of Perception and Cognition at the German Aerospace Center (DLR) on a topic that deeply inspired me.

I would also like to express my sincerest gratitude to my advisors, Wout Boerdijk and Maximilian Durner, for their patience and continuous support throughout the past 6 months. Their guidance and targeted feedback, both on an academic and technical level, were invaluable for the successful completion of this work.

On a more personal note, I would like to thank my family, Nikos, Antonia and Kostas, and my partner Andreas for their continuous love and support throughout the course of this master's program.

Abstract

Robust real-time unknown object segmentation is critical for the successful planning and execution of robotic manipulations in the real world. To further ensure collision-free path planning in dynamic, highly unpredictable environments, tracking techniques can be incorporated, resulting in temporally consistent mask predictions. Specifically semi-supervised tracking, where the algorithm is initialised with a set of target objects to be tracked, poses a particular interest for the robotic setup, due to its modularity and compatibility with tracking unknown objects. However, most available semi-supervised methods in the literature are not explicitly designed for tracking in cluttered, highly dynamic scenes. Hence, we propose a novel semi-supervised Deep Learning based tracker with a separate encoder for the past frames and a matching module for the past and current frame encodings, designed to alleviate the aforementioned challenges. For the past frame encoder, three novel options are examined: a siamese backbone encoder, a bounding box encoder and a mask encoder. Crucially, for the matching module two variants are proposed, a separable attention-based matcher and a novel GMM-based matcher. In the experimental part, the proposed options for the past frame encoder and the matching strategy are compared in view of their performance and computational cost. Moreover, a comparison of our method with both a simple heuristic-based and an established semi-supervised tracker is carried out. Finally, a qualitative analysis on the sim2real performance of the proposed method and the effect of tracking on the achieved segmentation quality is performed.

Kurzfassung

Eine robuste Segmentierung unbekannter Objekte in Echtzeit ist entscheidend für eine erfolgreiche Planung und Ausführung von Robotermanipulationen in der realen Welt. Um eine kollisionsfreie Bahnplanung in dynamischen, hochgradig unvorhersehbaren Umgebungen zu gewährleisten, können Tracking-Techniken eingesetzt werden, da sie zu zeitlich konsistenten Maskenvorhersagen führen. Insbesondere das semi-supervisierte Tracking, bei dem der Algorithmus mit einer Gruppe von zu verfolgenden Zielobjekten initialisiert wird, ist aufgrund seiner Modularität und Kompatibilität mit der Verfolgung unbekannter Objekte für robotische Anwendungen von besonderem Interesse. Jedoch sind die meisten in der Literatur verfügbaren semi-supervisierten Methoden nicht explizit für die Verfolgung in robotisch relevanten Szenen konzipiert. Daher schlagen wir einen neuartigen semi-supervisierten Deep Learning basierten Tracker mit einem separaten Encoder für die vergangenen Frames und einem Matching-Modul für die vergangenen und aktuellen Frame-Encodings vor, um die oben genannten Herausforderungen zu bewältigen. Für den Encoder der vergangenen Frames werden drei neue Optionen untersucht: ein Siamese-Backbone-Encoder, ein Bounding-Box-Encoder und ein Masken-Encoder. Für das Matching-Modul werden zwei Varianten untersucht: ein separable-Attention basierender Matcher und ein neuartiger GMM-basierter Matcher. Im experimentellen Teil werden die vorgestellten Optionen für den Past-Frame-Encoder und die Matching-Strategie im Hinblick auf ihre Leistung und Rechenkosten verglichen. Darüber hinaus wird ein Vergleich unserer Methode mit einem einfachen heuristischen und einem etablierten semi-supervisierten Tracker durchgeführt. Abschließend wird eine qualitative Analyse der sim2real Leistung der vorgestellten Methode und der Auswirkungen des Trackings auf die erreichte Segmentierungsqualität durchgeführt.

Contents

1. Introduction

Modern robotic research has shifted from designing robots solely for use in controlled industrial environments to equipping them for deployment in settings involving human-robot interaction. However, 'human' environments are inherently chaotic and highly unpredictable: Firstly, scenes encountered by robotic agents tend to be cluttered, as neither the background nor the object positioning are known and explicitly set up for robotic manipulation. Additionally, novel, unknown scenarios arise, as human operators change and interact with the environment. Moreover, scenes are highly dynamic, with both objects and the robot itself potentially in motion. In such a dynamic and unpredictable setup, robotic perception plays a crucial role in successfully planning and executing robotic manipulations.

In this context, dense object detection can be the first perception prerequisite for accurate object localization with 6D pose estimation (Sundermeyer, Marton, et al. 2018) or more robust grasp prediction (Sundermeyer, Mousavian, et al. 2021). Moreover, a class-agnostic operation of dense detection algorithms is critical for collision-free path planning in environments where unknown objects may be introduced to the scene at any time and should not be confused with the background. Most importantly, due to the dynamic nature of the scenes in such setups, time-consistent predictions are particularly important for a robust planning and execution of target manipulations. Additionally, temporally consistent predictions open up the possibility of new data generation for continuous learning during deployment, in a similar fashion to how human agents learn to detect new objects after careful inspection of multiple views in the course of time (Boerdijk et al. 2020).

Designing an algorithm with the aforementioned attributes, namely performing unknown, dense, time-consistent object detection in a robotic setup, is a particularly challenging task. Firstly, illumination changes and sensor noise in the camera input pose difficulties in accurate object prediction. An additional level of complexity is introduced from the dynamic nature of the scenes, due to the movement of the robotic system and/or the objects, potentially resulting in dramatic changes of the objects' appearance. Finally, another problematic factor for temporally consistent object recognition in real-world, uncontrolled environments is the level of 'clutterness' that may be encountered both with regard to how close the objects are positioned, but also due to objects being part of the background that need to be recognized for collision avoidance.

In recent years, important progress has been made in single-shot unknown object segmentation for robotic applications, with works that address the first two aforementioned aspects like (Durner et al. 2021) and (Kollar et al. 2022). However, extending such single-shot approaches

in the temporal domain is not trivial and remains an ongoing research topic in the field of robotic vision. Therefore, the main scope of this thesis is to address the problem of improving the robustness of unknown object instance segmentation in robotic applications from the view of temporal consistency by incorporating tracking techniques.

Tracking can be formulated as the task of assigning a unique id to a detected object in the first frame it appears and maintaining that id for the same instance in future frames in which it is successfully detected. Intuitively, this id association process for predicted objects in different frames can be based either on a correct notion about the trajectory each distinct object is following or an adequate model of its appearance (Bergmann et al. 2019). In practice, tracking techniques are often based on some form of information propagation between neighboring frames, either motion- or appearance-based.

Current established tracking approaches for unknown objects are not designed and optimized for the robotic setup. For example, the method proposed in (Athar et al. 2022) is designed to track a small number of large, distinct bodies, like the ones encountered in the popular DAVIS evaluation set (Pont-Tuset et al. 2017). In contrast, a robotic agent may be expected to detect and manipulate many smaller objects, e.g. tools, randomly scattered on a surface. Moreover, available methods are not optimally designed with regard to the computational cost, as they are not meant to be deployed on a robotic system. Therefore, they often feature elaborate attention mechanisms (Athar et al. 2022), (Kipf et al. 2022) or complex memory modules (H. K. Cheng et al. 2022), (Z. Yang, Wei, et al. 2021).

Motivated by the lack of a powerful and efficient tracking method for unknown object segmentation optimized for the challenges arising in robotic dynamic vision, this thesis aims at proposing a model with these characteristics. We draw inspiration from available works and propose a novel semi-supervised Deep Learning (DL) network, explicitly designed for performing temporally consistent object segmentation in particularly cluttered scenes that involve changes in an object's appearance caused by fast camera movement. Crucially, semi-supervised tracking allows for initialization of the process in a class-agnostic manner, solely based on binary segmentation masks of objects of interest, which is compatible with our problem formulation. Moreover, the initialization process can be realised in different ways, ranging from utilizing segmentation masks predicted by another network to human-in-the-loop interaction, thereby making this category of trackers highly modular. For the aforementioned reasons, we target a solution within the semi-supervised family of tracking techniques, especially optimized for the requirements of the robotic setup.

Our proposed semi-supervised network introduces temporal consistency in the predicted segmentation masks by matching past frame encoded data with the feature maps of the current frame. This process results in more temporally consistent feature encodings forwarded to the decoder. In this scope, we formulate the following research questions:

- How do different 'novel' mechanisms for past frame encoding and matching compare with respect to their performance and computational cost? Is there an optimal configuration for the robotic setup?

- Is our proposed method more successful in tracking unknown objects in cluttered scenes than available methods designed for a different domain?
- Besides the benefit of temporal consistency, does learned past frame information propagation also result in better segmentation quality compared to single-shot segmentation (i.e. segmentation that does not depend on information from past frames) in the context of unknown object segmentation?

This thesis is organized into 6 chapters. In chapter 2, the related work is presented with the main focus on available tracking methods in the literature. In chapter 3, the proposed DL architecture in all its variants is presented along with the theoretical background for the main modules comprising the model. In chapter 4, the experimental setup for the results and analysis included in chapter 5 are provided. Finally, in chapter 6 the derived conclusions from the experimental part and the outlook for future improvements are summarized.

2. Related work

In modern dynamic vision, tracking is encountered, directly or implicitly, in different problem formulations related to video understanding. For example, the tasks of Multi Object Tracking and Segmentation (MOTS) (Voigtlaender et al. 2019), Video Instance Segmentation (VIS) (L. Yang et al. 2019), Video Panoptic Segmentation (VPS) (Kim et al. 2020) and Video Object Segmentation (VOS) (Pont-Tuset et al. 2017) are closely connected and involve tracking as part of their objectives. All these problems tackle object detection in videos, but present differences in how the background class is treated or whether multiple instances of the same semantic class are encountered in the data. These distinctions in the problem formulation and objectives result in differences in the approaches and evaluation metrics introduced in the literature. However, in many cases inspiration can be drawn from solutions originally proposed for different problem paradigms.

As already mentioned in the introduction, the main goal of this thesis is to propose a DL model that predicts temporally consistent object segmentation masks in the context of robotics. Our class-agnostic problem formulation is closer to the VOS task, that is typically tackled with semi-supervised learning approaches. However, inspiration can also be drawn to some extent by concepts encountered in VIS and MOTS. For that matter, we aim at reviewing a wide range of algorithms and concepts that are available in the literature and may be relevant for a good solution for the task at hand.

To simplify the review process of existing works that may be relevant for designing a robust tracker in the robotic setup, three levels of categorisation are proposed. First, a distinction is made between **online** and **offline** tracking methods. For the **online** methods, that are more relevant to the task at hand, a distinction is also made between **modular** and **end-to-end**. Finally, online modular tracker are split into two subgroups, **heuristic based** and **learning based**. A summary visualisation of the categories involved in this review is provided in Fig. 2.1. It is worth noting that Fig. 2.1 does not involve all possible sub categories encountered in each group, but rather only the ones presented in this review and are most relevant to our problem.

2.1. Taxonomies of tracking methods

The first intuitive way to categorise tracking approaches is between **online** and **offline** approaches. **Offline** approaches typically receive as input a batch of frames which is available in advance. Offline methods typically demonstrate superior performance compared to online

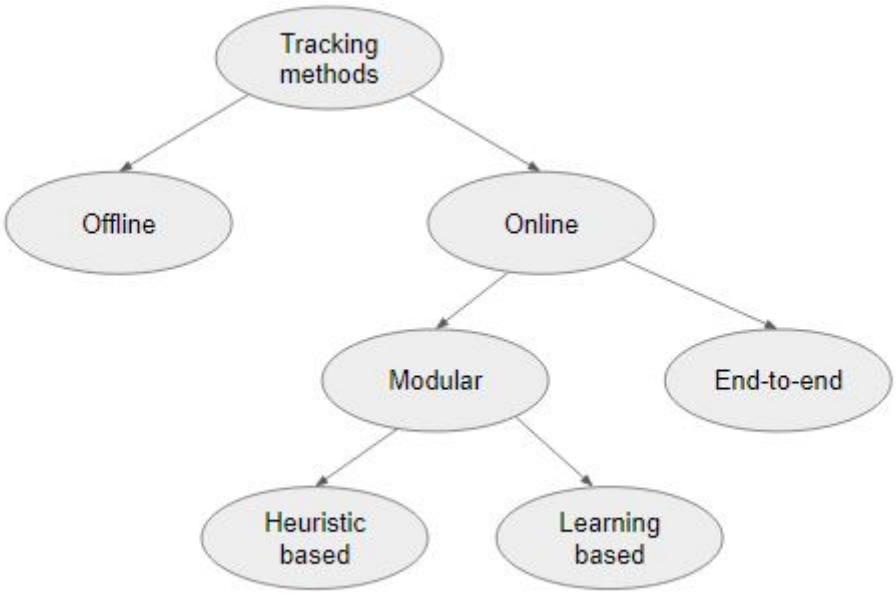


Figure 2.1.: Proposed taxonomies for tracking methods

methods, since optimization is taking place in both directions of time, as the model has access to data from both past and future frames. Offline methods though more powerful can only be used to analyse prerecorded video sequences and hence are not directly applicable in real time applications such as robotic tracking. In the case of **online** approaches on the other hand the input frames are handled in a sequential manner (Luo et al. 2021). This implies that the model can only learn from one or multiple past frames stored in a memory module. Therefore, online approaches are the only viable alternative for real-life applications like autonomous driving or robotic manipulation.

Online methods can be further split into two groups, **end-to-end** and **modular**, depending on how tracking is performed. In general, tracking can be seen as a combination of two subtasks: single shot object detection/localisation ignoring the aspect of time and time consistent object detection with id assignment (Luiten et al. 2020). End-to-end methods, typically learning based, perform detection and tracking jointly with a single module. Modular methods on the other hand, be it heuristic based or learning based, involve separate modalities in the tracking pipeline. For example, heuristic based modular methods tackle the two subtasks independently with two separate modules, one for single shot detection and one for detection association (tracking). Learning based modular trackers on the other hand typically require some initialisation for the tracking process. Hence, they feature a DL network that produces temporally consistent detections, but the inference process is initialised by a separate single shot object detector.

In the following sections, an overview of State-Of-The-Art (SOTA) methods belonging to each of the proposed categories is provided. Since the goal is to propose a solution for robotic tracking, more focus is posed on online methods and their subcategories. However, even a short overview of each subcategory can help understand trends in current research and draw inspiration for a solution that involves desired attributes.

2.2. Offline tracking methods

Most SOTA methods in offline tracking are transformer based. Notable examples of offline tracking methods that demonstrate superior performance on YouTube-VIS (L. Yang et al. 2019) are VisTR(Y. Wang et al. 2021) and its improvements DeVIS (Caelles et al. 2022) and SeqFormer (Wu, Jiang, et al. 2022). All three methods extend a transformer based object detection algorithm, either DETR (Carion et al. 2020) in the case of (Y. Wang et al. 2021) or deformable DETR (Zhu et al. 2020) in the cases of (Caelles et al. 2022) and (Wu, Jiang, et al. 2022) in the temporal domain. As expected, computing spatio-temporal attention is very computationally expensive, so (Caelles et al. 2022) and (Wu, Jiang, et al. 2022) introduce improvements like the use of deformable attention in (Caelles et al. 2022) or a more efficient way to handle object queries on a video level in (Xie et al. 2021). The computational complexity of attention mechanism is a problem that both online and offline methods suffer from.

As already mentioned in section 2.1, a significant advantage of offline tracking methods is that optimisation takes place in both directions of time during training. For example, in (Park et al. 2022) the authors present an offline version of STMs (Oh et al. 2019) by conducting both memory update and mask prediction on a clip (set of consecutive frames) instead of frame level achieving superior performance to STMs. In practice these algorithms can only serve loosely as source of inspiration for the solution to be designed. Realistically, learning from multiple frames in a batched fashion is not applicable in tracking for autonomous systems.

2.3. Online tracking methods

In the case of online tracking, more versatility is encountered among the methods that demonstrate SOTA performance. Examples of online approaches in tracking span from the simple but powerful SORT(Bewley et al. 2016) and DeepSORT (Wojke et al. 2018) to the transformer-based Trackformer (Meinhardt et al. 2022) and IDOL (Wu, Q. Liu, et al. 2022). However, the most powerful online methods achieving comparable performance with offline methods on challenges like YouTube-VIS (L. Yang et al. 2019) and MOTS20 (Voigtlaender et al. 2019) are DL based.

2.3.1. Heuristic based Modular Online methods

There is a wide range of heuristic based modular online tracking methods that decouple the detection from the tracking process. However, most heuristic based modular trackers involve some sort of motion or appearance assumption. For example, in the case of Kalman Filters (Kalman 1960) a linear motion assumption is made. Another common modular approach may involve some sort of matching based on appearance similarity like the Intersection over Union (IoU) association baseline used in (Weber et al. 2021). In IoU association, a unique track id is initialised for each detected object in the first frame and this id is propagated in the next frames with a linear sum assignment algorithm like the Hungarian method based on IoU overlap scores between detections. Other examples of modular methods may involve a Markov decision process like the method proposed in (Xiang et al. 2015) or some sort of optical flow warping (Teed et al. 2020) of predicted masks from $t-1$ in t combined also with an IoU association/ matching step (Weber et al. 2021).

A particularly interesting modular tracker is SORT (Bewley et al. 2016), a Kalman-based tracking method with a data association step. SORT can be combined with any probabilistic detector, as it uses confidence scores apart from bounding boxes to estimate the position of the tracked object in the next frame. The method owes its popularity to its relatively high performance on the MOT challenge (Milan et al. 2016) despite its simplicity. SORT is often used as baseline for more complex learning based approaches like Yan et al. 2022, which have long outperformed heuristic based tracking in all important evaluation benchmarks.

2.3.2. End-to-end Online methods

The term end-to-end refers to methods that entail some sort of modification or further training on the detector they build upon. In the literature, a variety of end-to-end tracking algorithms is encountered, like the LSTM-based tracker introduced in (Gordon et al. 2017) or the transformer-based Trackformer (Meinhardt et al. 2022).

One quite dominating concept in converting transformer-based detectors into end-to-end trackers is the concept of track queries. In Trackformer (ibid.) the authors adjust deformable DETR (Zhu et al. 2020) for tracking by making use of the concept and report very good results on the MOT17 (Milan et al. 2016), MOT20 (Dendorfer et al. 2020) and MOTS20 challenges (Voigtlaender et al. 2019). In (Zhan et al. 2022) the authors adapt the powerful MaskFormer (B. Cheng et al. 2021) and achieve very good results on the VIS benchmarks OVIS (Qi et al. 2022) and UVO (W. Wang, Feiszli, et al. 2021). The notion of propagating high-level information from detections of the previous frame is not solely encountered in supervised end-to-end trackers that work in (Meinhardt et al. 2022) and (Zhan et al. 2022). In a way, the concept of 'slots' proposed in (Kipf et al. 2022) and the high-level features in (Athar et al. 2022) act for the model similarly to 'track' queries propagating previous information on a detection level. This is in contrast to information propagation performed on a feature map level from the previous frame preferred in memory-based methods like (Oh et al. 2019) and (Z. Yang, Wei, et al. 2021), which is in general more computationally intensive.

A notable transformer-based online tracker is IDOL (Wu, Q. Liu, et al. 2022). IDOL uses contrastive learning to perform tracking by association based on appearance similarity. The use of a contrastive loss enables the model to learn more discriminative embeddings for association, achieving comparable results to offline trackers. A less resource demanding transformer-based architecture is Unicorn (Yan et al. 2022). In Unicorn, a matching mechanism with deformable attention is used to capture interactions between neighboring frames. Matching mechanisms as a concept are present in many works on online tracking and are very effective in propagating information from previous frames. They can be applied on a feature map level like in (Oh et al. 2019) or on higher order descriptors (generated from the ground truth) like in (Athar et al. 2022).

Most end-to-end methods encountered in the literature are supervised learning models designed to track many instances of the same semantic class, like cars or pedestrians. This attribute of end-to-end online methods is not compatible with our class-agnostic problem formulation. Therefore, a different category of online learning based methods is also reviewed, to help us propose a solution that better accommodates the problem's requirements.

2.3.3. Learning based Modular Online methods

An alternative to end-to-end learning based online methods, which are typically supervised, are semi-supervised tracking approaches. Unlike supervised methods, semi-supervised trackers require an initialisation in the first frame, either via the ground truth annotations for the objects to-be-tracked or via the predicted objects from another network. Hence, semi-supervised are modular by their nature, as they can be easily combined with different object detectors for initialization without further adjustment.

An important semi-supervised, learning based modular online tracker is the memory based approach proposed in Space-Time Memory networks (STM)(Oh et al. 2019). The memory based approach in STM involves storing past encoded feature maps in a memory module. Then the stored past feature maps are utilised in learned spatio-temporal matching with the current encoded feature map to produce the current frame predictions.

The idea of storing extracted features from multiple past frames in a memory module to boost prediction performance on a video sequence first introduced in (ibid.) has been very influential for the current SOTA semi-supervised methods. For example, AOT and its variants (Z. Yang, Wei, et al. 2021), (Z. Yang, Miao, et al. 2022), (Z. Yang and Y. Yang 2022) propose a more efficient way to perform multi-object matching and segmentation. Instead of performing matching and decoding for each mask (target) in parallel, they propose an efficient way to project and associate multiple objects in the same high dimensional embedding space. On a different note, XMem (H. K. Cheng et al. 2022) leverages a more elaborate memory mechanism with multiple independent memory modules, thus achieving superior performance in longer videos compared to STM and its variants.

The concept of learned spatio-temporal matching, typically with an attention mechanism, has been very influential also for more lightweight online methods that omit the memory module and only match with information from a single past frame like (Kipf et al. 2022), (Athar et al. 2022). These methods also distinct themselves from memory based approaches since they propagate past information on a ground truth/prediction instead of feature map level. More specifically, the authors in (Kipf et al. 2022) compare a variety of different cue sources from the first frame ground truth (center of mass, bounding box, segmentation) and demonstrate that even the most naive motion cues from the first frame - such as the center of mass of detected objects suffice for successful tracking of objects in the next frames. On the other hand, in (Athar et al. 2022) the authors demonstrate that even just high-order descriptors from the first frame detection (the whole segmentation mask instead of feature maps) can help to guide the model for association consistency in upcoming frames.

3. Method

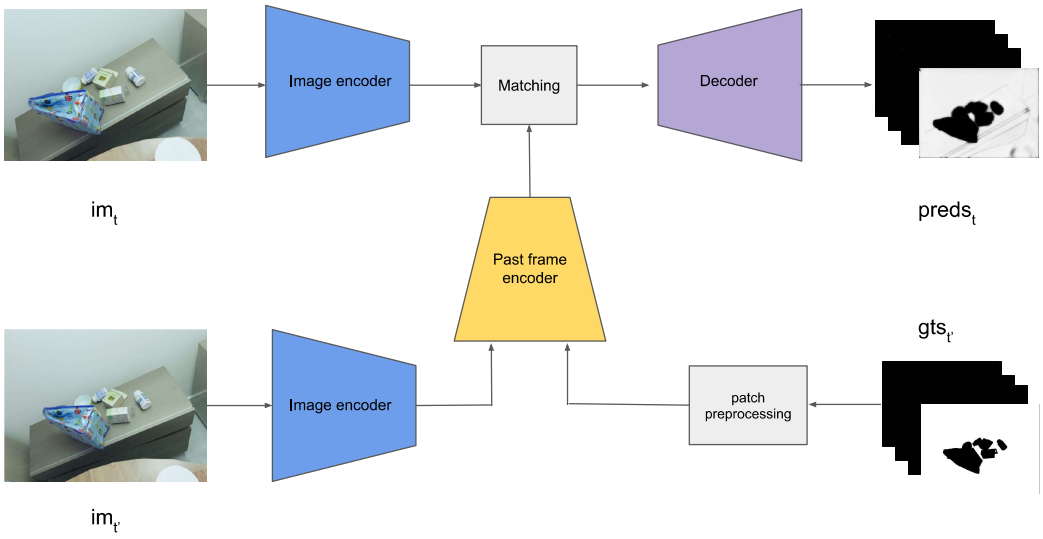
As already stated in the previous chapters, the proposed approach is an online, modular, semi-supervised, DL based tracker. Single shot unknown object segmentation networks often follow an encoder/ decoder structure. Semi-supervised DL based trackers on the other hand typically require more sophistication in the model architecture. The proposed method comprises four key components: an image feature encoder, a decoder, a past frame encoder and a matching module. Crucially, a matching mechanism is utilised to estimate similarity between encodings from previous and the current frame and to feed the decoder with more temporally consistent encodings that combine information from the previous and current frame.

A short description of the role of each module encountered in the proposed architecture follows:

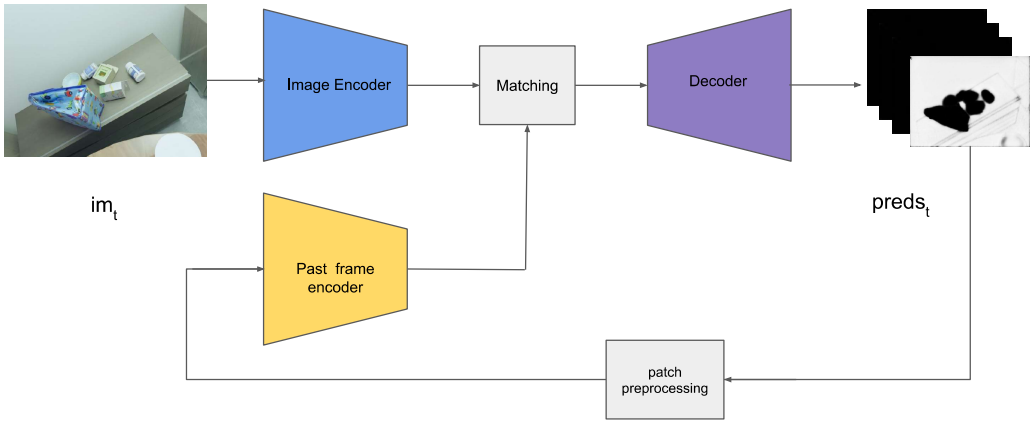
- The **image feature encoder** extracts feature representations of the current frame.
- The **past frame encoder** encodes past frame information to ensure temporal consistency. In some model variations past information is obtained by combining high level information from ground truths/predictions and previous feature maps encoded with the image feature encoder.
- The **matching block** matches the past frame features and current feature map in a more informed and temporally consistent feature map.
- The **decoder** decodes the processed feature map into binary segmentation masks (one binary mask for each tracked object and the background).

A modular overview of the proposed method is provided in Fig. 3.1. During training, tracking is initialised based on the ground truth masks from a past frame. During inference, initialisation can be either performed via a provided past frame ground truth or predictions from another DL network. After initialisation and for the rest of the sequence length, past information is propagated via the predictions from the preceding frame.

In the following sections, more details about the sub-modules encountered in the proposed architecture are provided. For the image feature encoder a transformer based backbone was preferred and not further ablated. For the rest of the modules, different options were investigated and compared with regard to their performance and computational cost in chapter 5.



(a) Model overview using past frame ground truths for initialisation



(b) Model overview using past frame predictions for initialisation

Figure 3.1.: Model overview with different initialisation types

Note that t refers to the current frame and t' to the past frame in a sequence of total length T

3.1. Model overview

3.1.1. Image feature encoder

The image feature encoder block converts an RGB image input to a set of multi-dimensional feature maps. It is used to encode the current input image frame, but also the past image frame for some variants of the past frame encoder. Theoretically, any standard image processing backbone, CNN or transformer based, can potentially be used as image feature encoder. In practice, the SegFormer-B3 (Xie et al. 2021) backbone was preferred, due to its superior performance with less learnable parameters compared to the popular ResNet-101 (He et al. 2016).

The SegFormer-B3 belongs to the broader category of transformer based backbones, first introduced by (Dosovitskiy, Beyer, et al. 2021). Transformer-based backbones generally lack the inductive biases present in CNNs and require longer pre-training to converge to meaningful representations (ibid.). However, the more elaborate variants of transformer based backbones present superior performance to CNNs for example by following a hierarchical structure in the encoder like in the case of the SegFormer (Xie et al. 2021) or a hierarchical design with a shifted window strategy like in the case of the Swin family Transformers (Z. Liu et al. 2021). A short description of the SegFormer-B3 backbone encoder follows.

The input image of size H × W × 3 is first divided into patches of size 4 × 4 instead of 16 x 16 like in the case of ViT. This designer choice is generally more suitable for the dense prediction task of video object segmentation (Dosovitskiy, Beyer, et al. 2021). Then the resulting patches are led to the hierarchical Transformer encoder resulting to multi-level features at different scales of the original image size 1/4, 1/8, 1/16, 1/32. Depending on the type of matching performed in the matching module, all resulting intermediate feature maps may be involved or just the final output of the encoder at scale 1/32.

The transformer block used repeatedly in the encoder consists of three sub modules: an Efficient self-attention block, a Mix-FFN (Feed Forward Network) block and an Overlap Patch Merging block. Efficient self-attention reduces the complexity of the attention mechanism and thus renders attention computation on large image resolutions feasible. The Mix-FFN block is proposed as a data-driven positional encoding mechanism, combining a 3 × 3 convolution and an Multilayer Perceptron (MLP) into each FFN block. Positional encoding mechanisms are typically used along with attention mechanisms to retain positional information of the sequence, which is lost after the attention operation. Finally, Overlapped Patch Merging has a similar effect to a pooling operation and is used to shrink hierarchical feature maps.

Efficient attention was first introduced in (W. Wang, Xie, et al. 2021) and involves a spatial reduction operation that reshapes the input sequences to sequences of smaller input size, resulting in less attention operations. The formulas for efficient attention are provided in (3.1), (3.2) and (3.3) where K the keys, Q the queries, V the values and d_{head} the hidden dimension same as in classic attention mechanism introduced in (Vaswani et al. 2017). All K, Q, V are tensors of shape $\mathbb{R}^{NxC}$ where $N = HxW$ is the length of the sequence equal to the flattened

patch spatial dimensions. The length of the sequences N is reduced by applying a reduction rate R^2 to all three K, Q, V as indicated for K in (3.2). The original tensors can be recovered by applying (3.3).

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_{head}}})V \tag{3.1}$$

$$K' = Reshape(\frac{N}{R^2}, CR^2)(K) \tag{3.2}$$

$$K = Linear(CR^2, C)(K') \tag{3.3}$$

For more details on the Mix-FFN and the Overlapped Patch Merging blocks we refer the reader to (Xie et al. 2021).

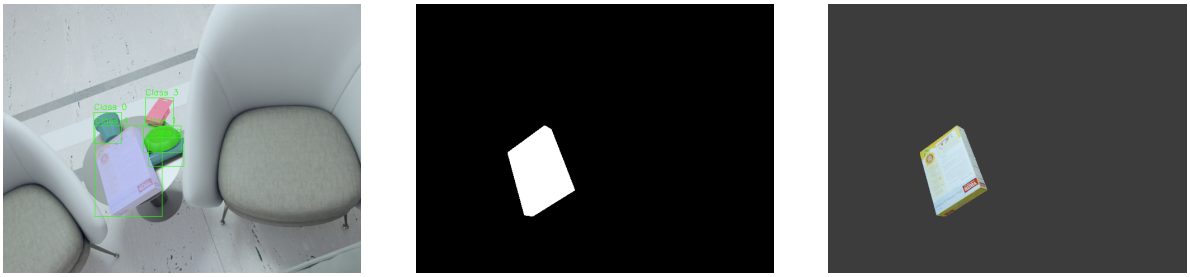
3.1.2. Past frame encoder

In the course of this work, three main variants for the past frame encoder are examined. In the first case, the backbone past frame encoder is built on an identical SegFormer-B3 block with the image feature encoder with different weights. The two other options are more lightweight alternatives to the 'backbone' image encoder, the motion-based 'bounding box encoder' and the appearance-based 'mask encoder'. A more detailed description of the three examined encoder options follows.

Backbone encoder

In the first past frame encoding case, the encoder design is very similar to the image feature encoder. More specifically, a siamese SegFormer-B3 encoder is used with different weights than the image feature encoder SegFormer-B3. The past frame encoder inputs result from masking a past input image with either the ground truth (during training) or the predicted binary masks from the same previous frame (during inference). The result is an RGB image that involves only the highlighted object of interest, see Fig. 3.2. The resulting patches are then downsampled from 480x640 to 120x160 with bicubic interpolation. An overview of this model variant is provided in Fig. 3.3.

In general, this past frame encoding approach results in a rich feature representation of the incoming information from the previous frame which is expected to be reflected in the performance. The downside however is that this past frame encoder option results in a bottleneck both in training and inference, as it requires training two identical SegFormer-B3 that do not share weights, one for encoding the current frame input and one for encoding the patches. Moreover, the very nature of the patches led to the encoder may be sub-optimal for guiding the network to track objects of interest, as they involve many black (background) pixels which are encoded along with the non-trivial masked object pixels and used in the



(a) Example overlay (b) Example binary mask (c) Example patch

Figure 3.2.: Patch generation for backbone encoder

matching process. This shortcoming of the backbone encoder is addressed in the mask encoder variant, where only features relevant to a respective class are involved in the matching process.

Bounding box encoder

A more lightweight variant of past frame encoder investigated in this work is bounding box (BB) based. Similar to a type of 'slots' proposed by Kipf et al. 2022, the idea of using bounding boxes to generate and encode past frame data is explored. The intuition behind using bounding boxes to carry information about the motion of the objects in the scene is that they play a similar role to positional encodings. Using bounding boxes results in a more minimal representation of the previous frame information in the latent space. More specifically, for each predicted cluster we end up with a single 256-dimensional feature representation instead of multi-feature representation [n_{feat} , 256] resulting from a single resolution mask encoder. However, even this very minimal representation is expected to successfully guide the model in non-challenging sequences with not many occlusions or appearance changes.

Past frame encoding generation from bounding boxes requires converting binary masks from ground truth (during training) or predictions (during inference) to bounding boxes. Binary masks can be easily converted to bounding boxes by finding the coordinates of pixels at the masks boundary $x_{min}, x_{max}, y_{min}, y_{max}$. Special care is required in case the predictions are involved in the past encoding generation, which is typical during inference. In that case, predictions are converted to binary masks with an $argmax$ operation and subsequently the bounding boxes are obtained from the non-trivial mask predictions.

The obtained bounding boxes are encoded with a simple MLP with one hidden layer and a ReLU non linearity. Input size is 4 and output of the MLP is 256 to match the latent size of the current input image feature map with which matching is performed with. A more detailed overview of the model with a BB encoder is provided in Fig. 3.4.

Mask encoder

The third variant of the past frame encoder involves combining both high order and low level descriptors from the past frame, combining elements encountered in (Oh et al. 2019) and (Athar et al. 2022). More specifically, the input image from a neighboring (past frame) is encoded with the same image feature encoder used for the current input. This results in a rich low level feature map representation of the past frame. Subsequently, the extracted feature maps in different spatial scales (1/4, 1/8, 1/16, 1/32) are masked with the downsampled past frame binary masks, that are obtained with k-nearest interpolation. Since downsampling can result to all zero masks in smaller scales, which has proven to be problematic for cluster initialisation in GMM matching, a projection mechanism for pixels of non-negligible masks in the original scale to the small mask scale is utilised to deal with the issue. An overview of the feature map masking operation at different feature map resolutions is provided in Fig. 3.5.

Apart from expanding the masking operation from (ibid.) to multi scale feature maps, the mean operation applied after masking the feature maps is omitted. This results in assigning each high-dimensional ($h_{dim} = 256$) feature of the feature maps to a cluster corresponding to each class (object) present in the previous frame. It is worth noting, that the resulting clusters for each class-separate object in the input image typically have different sizes [n_{feat}, 256] and cannot be stacked along the n_{cl} dimension of predicted classes forming a tensor of shape [n_{cl}, n_{feat}, 256]. Instead they are stored in memory as a nested list with length equal to the total number of objects/classes in the frame n_{cl}. The stored clustered multi-scale features are then passed on to the matching module to propagate past information in the current frame.

The authors in (ibid.) also propose splitting the background in multiple patches in a gridwise fashion, that results in multiple background descriptors that are closer to the object descriptor sizes. This option to include multiple background descriptors was also considered interesting to investigate and was further extended to learning multiple descriptors for the tracked objects.

3.1.3. Matching

The matching module is the core network component where a similarity computation between the past and current features is taking place, enabling time consistent predictions. For the matching process, two novel options are examined, a separable attention-based matching process and a clustering-based matching technique with Gaussian Mixture Model (GMM)s. A detailed description of both variants follows.

Separable attention matching

One way to match the information in the encoded feature map and the past encoded features is via attention. More specifically, MultiHeadAttention with $nheads = 8$ is adjusted into a

novel version of Separable Attention. The roles of queries Q, keys K and values V in this self-attention computation are assigned as follows:

- Q: current image encodings
- K, V: past feature encodings

Before separable attention computation between Q and K, V, both tensors are normalised with a LayerNorm operation. Then, the softmaxed inner product between Q and K is computed according to:

$$A = Softmax(\frac{QK^T}{\sqrt{d_{head}}}) \tag{3.4}$$

. Subsequently both the softmaxed inner product of Q, K indicated with A and the values tensor V are split in n_{cl} tensors, so that each tensor A_i, V_i only involves features assigned to a distinct class/object i present in the previous frame. Finally, the original attention inner product mechanism (Bahdanau et al. 2014) is computed separately for each pair of A_i, V_i tensors corresponding to a separate class/object in the frame according to:

$$SeparableAttention_i = A_i V_i \text{ for all i } \in n_{cl} \tag{3.5}$$

. The intuition behind using attention as a matching mechanism can be shortly described as follows: the softmaxed inner product A from (3.4) serves as a similarity measure between past and current frame features. Then, for each cluster $i \in n_{cl}$, the relevant past features of the cluster V_i are involved in the decoding process based on the measure of their similarity to current frame features A_i. This operation results in temporally consistent features relevant for both frames being led to the decoder. Crucially, attention computation is optimised during training, since all Q, K, V embeddings undergo a linear transformation that includes learnable weights before involved in seperable attention computation according to (3.4) and (3.5).

GMM matching

A more lightweight alternative to separable attention is a matching strategy based on GMMs. GMM is a probabilistic clustering method and can be used out of the shelf to perform semantic segmentation in an unsupervised manner. GMMs have also been used to improve the class estimations of deep neural networks in the context of semantic segmentation by extracting global representations (clusters) from the extracted feature map (Liang et al. 2022), (Saire et al. 2022). Drawing inspiration from works like the aforementioned a GMM based feature matching strategy is devised.

The core assumption in GMM modeling is that all the available data can be modeled as a simple linear superposition of Gaussian components (Bishop 2006). More specifically, to each assumed cluster k is assigned a unique Gaussian component. The Gaussian component parameters mean μ_k and variance Σ_k, the prior probabilities for each cluster π_k and the probabilistic cluster assignments for each datapoint, known as responsibilities $\gamma(z_{nk})$, are typically estimated with the EM algorithm in an iterative fashion (ibid.).

The EM algorithm comprises of two steps repeated in every iteration:

- In the Expectation step (e-step) the probability of each datapoint x_n belonging to each of the assumed K clusters (responsibility) $\gamma(z_{nk})$ is estimated based on the assumed Gaussian component parameters μ_k, Σ_k and π_k.
- In the Maximization step (m-step) the mean and variance for each assumed Gaussian component μ_k, Σ_k as well as the prior probability of belonging to the component π_k are updated based on the current cluster assignments $\gamma(z_{nk})$.

Due to its iterative nature, the EM algorithm requires an initialisation for the values of μ_k, Σ_k, π_k. This initialisation can be either random or informed, for example using a non-probabilistic clustering algorithm like K-means. In our use case, the initial mean and covariance values for each Gaussian distribution are estimated in combination with the mask encoder variant for past frame encoding. More specifically, the encoded past frame features (K, V) are assigned to a cluster after a masking operation is performed with the past frame's binary masks. This initial cluster assignment of the past frames is used to initialise the EM algorithm.

Algorithm 1 EM algorithm for matching

Initialise hyperparameter n_{iters} ▷ n_{iters}: num of EM iters
Set x = kvs, K = len(kvs) ▷ K: num of clusters
Initialise $\mu = mean(x), \Sigma = covs(x), \pi = 1 \in R^K$
while $i < n_{iters}$ **do**
    E-Step: $\gamma(z_{nk}) = \frac{\pi_k N(x_n|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} N(x_n|\mu_j,\Sigma_j)}$)
    M-Step:
    $\mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n$
    $\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k^{new})(x_n - \mu_k^{new})^T$
    $\pi_k^{new} = \frac{N_k}{N}$
    where $N_k = \sum_{n=1}^{N} \gamma(z_{nk})$
    $i \leftarrow i + 1$
end while
Set x = q, $\mu = \mu^{new}, \Sigma = \Sigma^{new}, \pi = \pi^{new}$
E-Step: $\gamma(z_{nk}) = \frac{\pi_k N(x_n|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} N(x_n|\mu_j,\Sigma_j)}$)

An overview of the matching process with the EM algorithm is provided in Algorithm 1. After initialisation, a fixed number of iterations n_{iter} is performed resulting in an improved cluster estimation μ_k^{new}, Σ_k^{new}, π_k^{new} based on the past frame features. Subsequently, the e-step of the EM algorithm with the cluster estimations from the past frame is also executed on the current frame features resulting in a time consistent cluster assignment for the features of the current frame. For a more expressive matching, similar to separable attention we can initialise multiple GMM heads resulting in an even richer representation.

After the EM algorithm execution on the current frame features with the cluster initialisation obtained by the EM algorithm run on the past features, the responsibility of each low resolution feature map belonging to a cluster is obtained. This pixel-level probabilistic cluster assignment can be either further decoded with a DL decoder with learnable parameters like an FPN or an MLP or can be directly upsampled and used as a class estimate on a pixel level in the original input image resolution. More details about the decoding process are provided in the following subsection.

3.1.4. Decoder

After the matching step, the multi-scale matched feature map is led to the decoder block. In this work, three options for the decoder module are included in the comparisons: the FPN decoder (Lin, Dollár, et al. 2017), the MLP decoder proposed by Xie et al. 2021 and a responsibility upsampler only relevant for GMM matching.

FPN decoder

Depending on the type of matching, only the smallest resolution features (single res) may have undergone through the matching process or all of them (multi res). The role of a classic CNN-based decoder is to process the multi-scale features with convolutions and interpolations and obtain the final prediction map in the original image shape. The default preferred CNN decoder is a feature pyramid network (FPN) (Lin, Dollár, et al. 2017), a rather popular choice for the task of instance segmentation. FPNs can be combined with any hierarchical feature extractor that produces feature maps at several scales, like most CNN backbones or any hierarchical transformer based backbone, e.g. the SegFormer. FPNs achieve superior performance by enhancing higher resolution features in the top-down pathway (decoder) with features with the same spatial size from the bottom-up pathway (encoder) via lateral connections. In this work, the bottom-up features led to the decoder are the matched feature maps in all the resolutions they are available. In case of single scale matching, in the scales matched features are not available, the original input image encoder features are instead fed to the decoder as skip connections.

MLP decoder

The All-MLP decoder proposed in SegFormer (Xie et al. 2021) is also included in the experiments as a powerful alternative to the CNN based FPN decoder with the pyramid structure. The all MLP lightweight decoder from (ibid.) consists of four submodules. First, all encoded feature maps in different scales are led to an MLP to obtain an output with a unified channel dimension. In the second step, the features are upsampled to 1/4 of the original image input scale. Finally, a dense prediction map is obtained after two more fully connected layers in 1/4 of the input scale. For more details, the reader is referred to the original SegFormer paper.

Responsibility upsampler

In case some combination of the mask encoder with gmm matching is used, a more interpretable alternative to the FPN/MLP decoder can also be utilised. Instead of using a CNN decoder with additional learnable parameters, there is also the option to directly upsample the estimated cluster responsibilities for the feature maps in different scales to the original image scale. The feature map upsampling is performed with bicubic interpolation. The resulting probabilistic cluster estimations in the original image scale can be directly converted to binary masks by applying an *argmax* operation.

The most important advantage of the responsibility upsampler compared to other CNN or MLP based decoders is that the obtained output is more interpretable. Both the FPN and MPL decoders upsample the low scale feature maps with operations involving learnable weights that are optimised based on a loss function and are hard to explain on a parameter level. Most importantly, for the purpose of loss computation the predicted values are bound between [0,1] with a softmax operation, which however does not correspond to a real probability. In contrast, the output of the responsibility upsampler can be directly interpreted as the probability of a pixel belonging to a cluster (object class), since the only operation between the low scale feature map responsibilities, which are corresponding to the probability of a high dimensional feature belonging to a cluster (object class) and the final per pixel class prediction is an interpolation. The result is a probabilistic dense prediction which is easier to interpret.

3.2. Loss function

The preferred training loss is the Cross Entropy loss, a popular choice for the task of object segmentation when pixel level annotations are available for all classes including the background. Cross Entropy was first popularised in the domain of medical imaging by Ronneberger et al. 2015 and is broadly used for other tasks as well.

In cross entropy loss, the total loss is computed as the sum of multiple loss terms involving all N pixels of a binary mask and all channels C equal to the number of predicted classes in the image including the background class $c = 0$. More specifically, the loss is computed based on the log probability of each pixel belonging to each class $\hat{y}_{ic}$ and the respective term y_{ic} from the one-hot encoded ground truth for each pixel according to (3.6). The probability estimations $\hat{y}_{ic}$ are obtained after applying the softmax operation to the network's output. In the ideal case, if predictions on a pixel level are optimal, the term $\hat{y}_{ic}$ for the correct class c will be close to 1, hence $log(\hat{y}_{ic})$ will be close to zero. Since all other terms $\hat{y}_{ic}$ for a particular pixel i will be also zero from the ground truth, the loss will be minimised. In that way, Cross Entropy loss has the capacity to give very accurate mask predictions.

$$Loss = -\sum_{t}^{t}\sum_{i}^{N}\sum_{c}^{C} y_{tic} log(\hat{y_{tic}}) \tag{3.6}$$

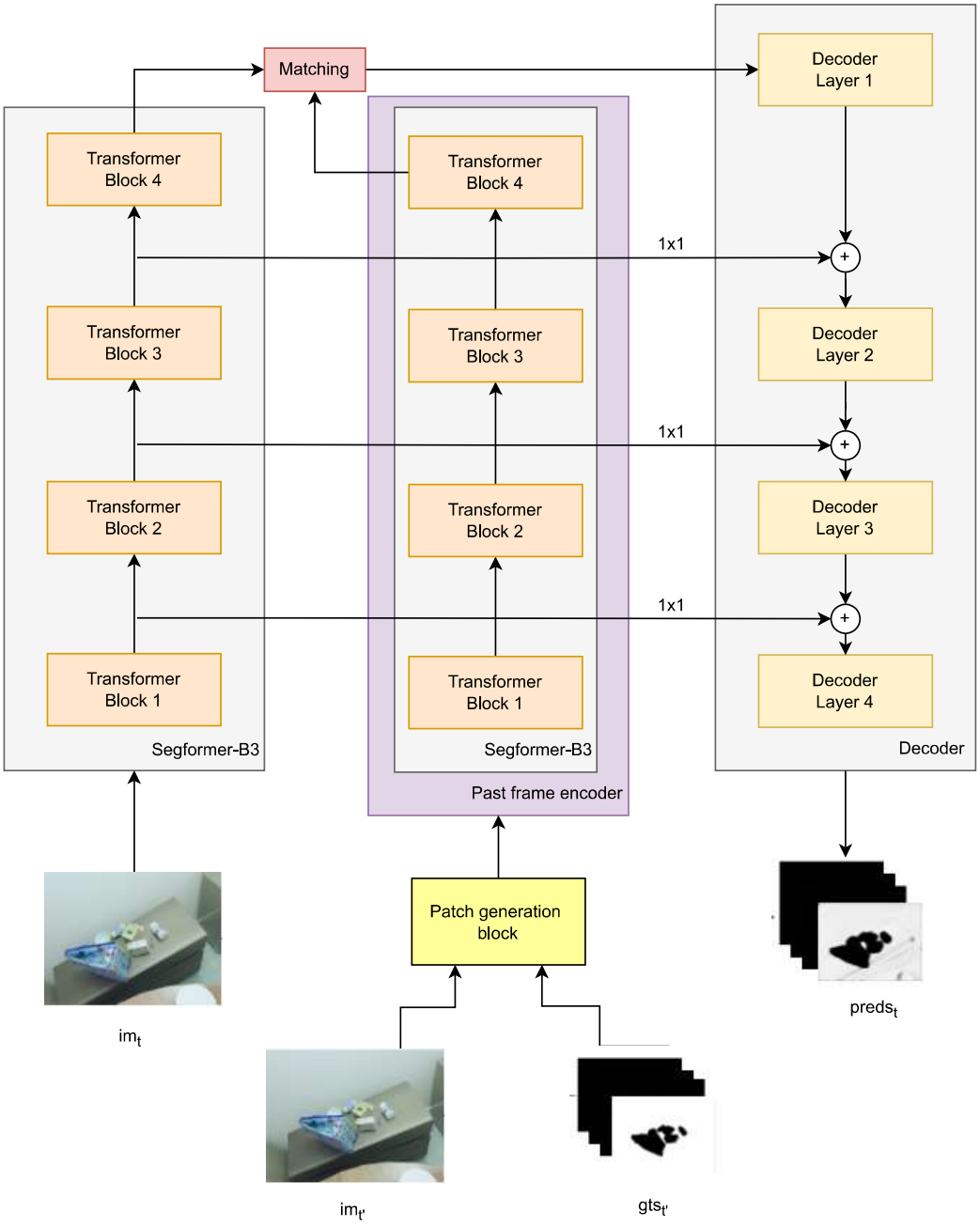


Figure 3.3.: Detailed model overview of the proposed backbone encoder

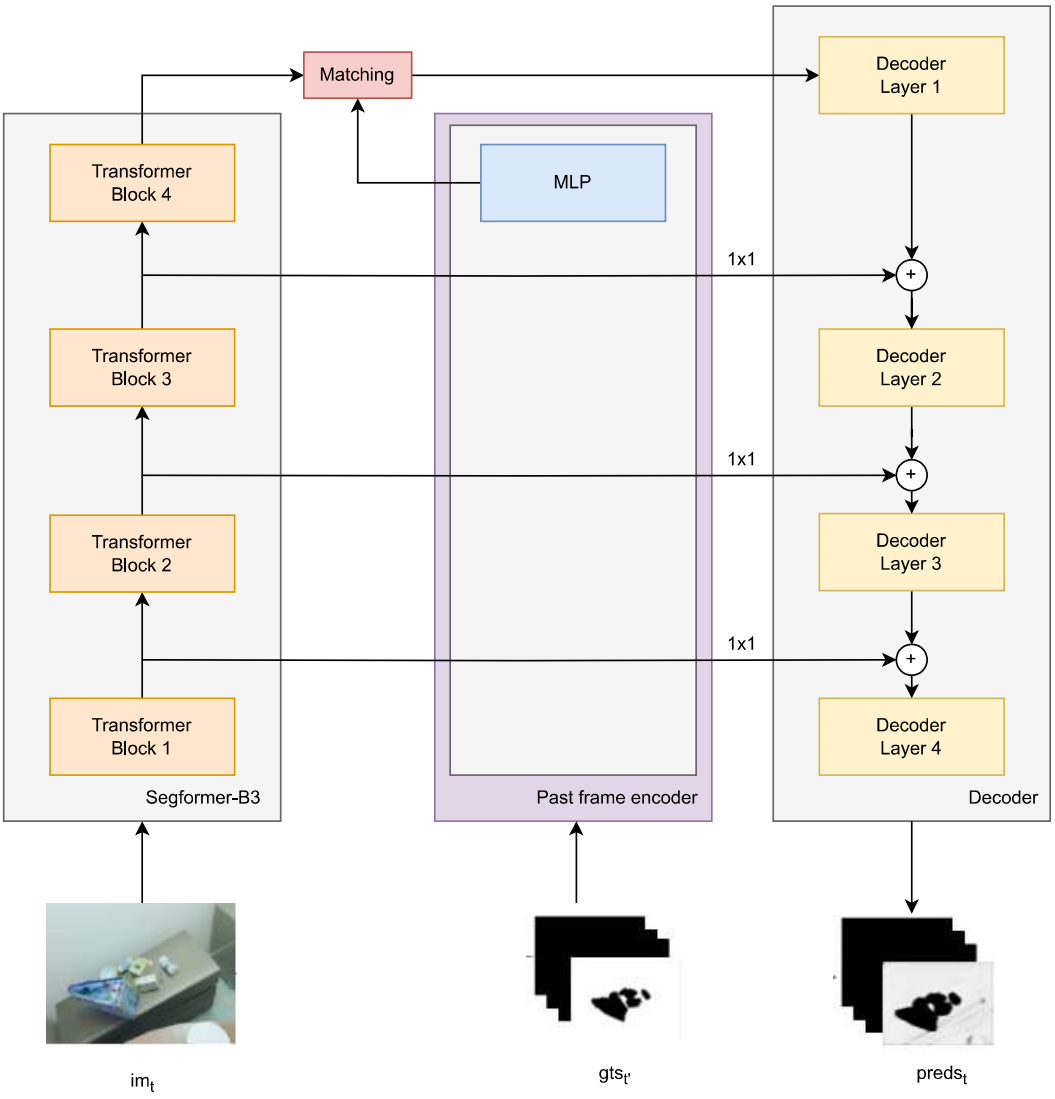


Figure 3.4.: Detailed model overview of the proposed bounding box encoder

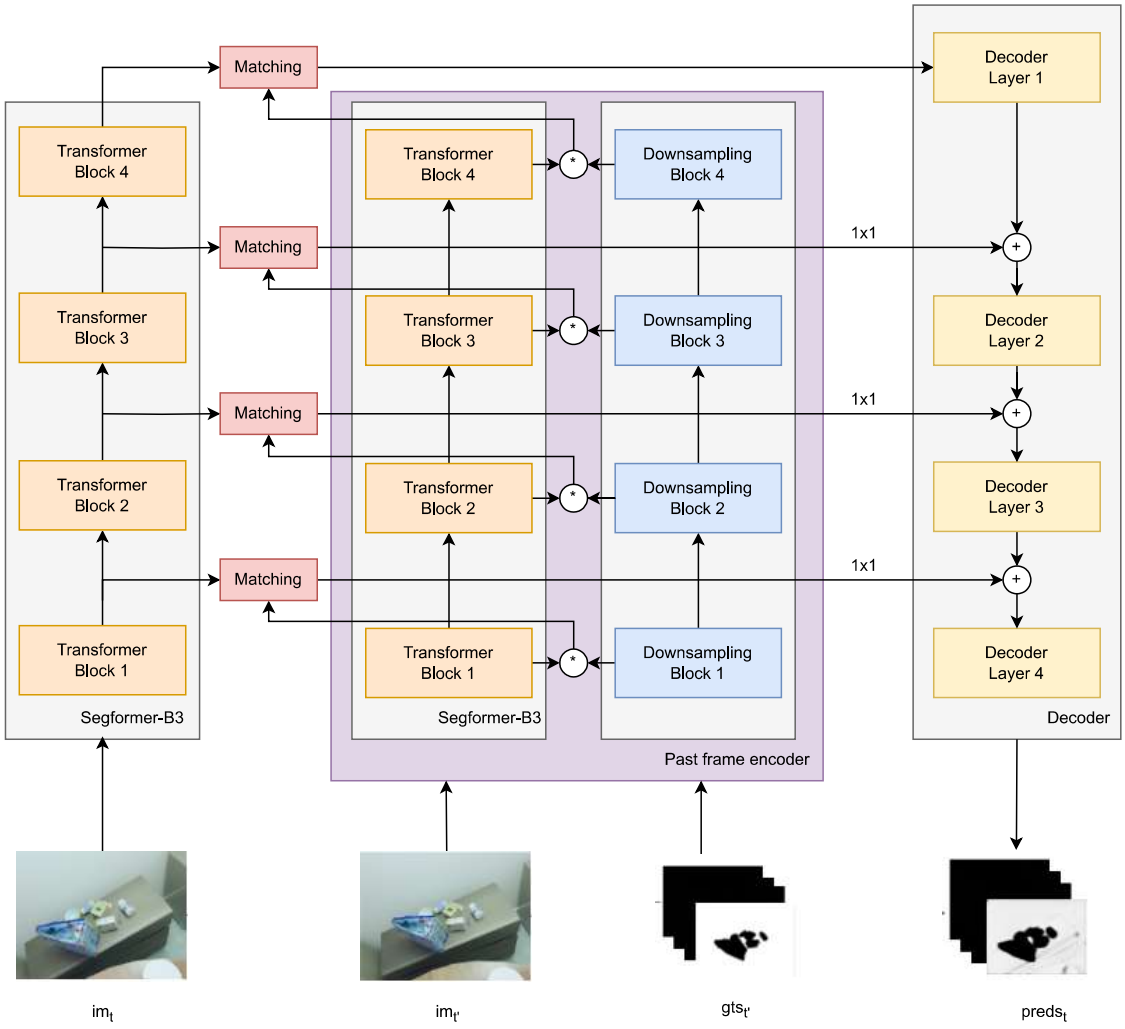


Figure 3.5.: Detailed model overview of the proposed mask encoder

Note that the * operator is indicating a masking operation that results in feature cluster assignment

4. Experimental setup

In this chapter, details on the experimental setup for the upcoming analysis in chapter 5 are presented. First, the training dataset and the main training setup for all model variants proposed in this work are discussed. Then, the evaluation datasets and metrics involved in the method comparisons are described in short. Finally, the semi-supervised tracker HODOR (Athar et al. 2022) and the unknown object segmentation algorithm INSTR (Durner et al. 2021) are shortly presented to help the reader understand the details behind experiments in chapter 5.

4.1. Training dataset

The dataset used for training is a synthetic dataset created with BlenderProc (Denninger et al. 2019). It consists of scenes of 10 subsequent frames from different vantage points resulting from a simulated camera movement. During dataset generation, the initial and final camera coordinates are sampled randomly from a range of possible values. Once the initial and final camera position are defined, a trajectory is formed and the 10 subsequent frames are sampled linearly along the path formed by the camera trajectory. Depending on the camera motion, objects may be partially occluded by other neighboring objects or may completely disappear between frames.

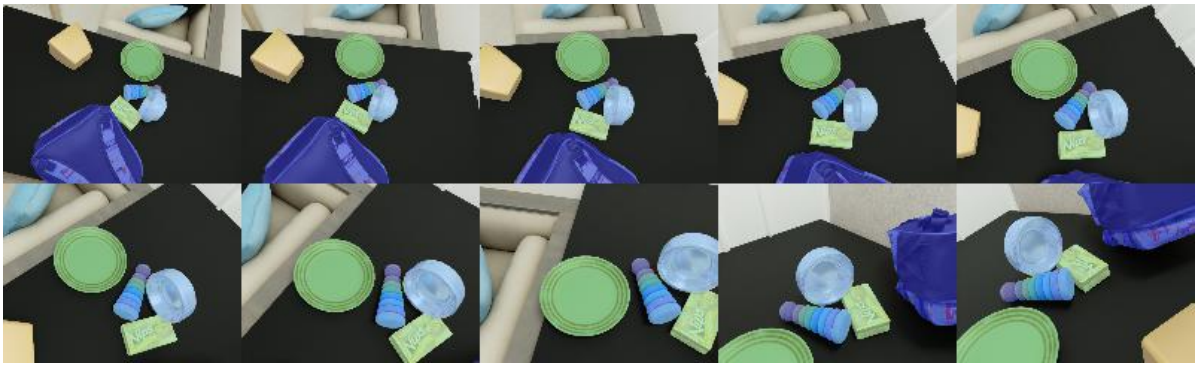
The dataset includes 989 sequences. In total, a number of 5 to 12 objects are present in each scene, similar to the dataset used to train INSTR (Durner et al. 2021), the method serving as our single shot unknown object segmentation baseline. In practice, BlenderProc scenes are generated by simulating a free fall of a number of objects sampled from the database on a table. As a result of the object free fall, some small objects may be completely hidden for the whole sequence if the objects end up in a pile. Therefore, in practice, there are cases where the model is trained in scenes with less than 5 objects visible.

Most similar semi-supervised learning methods in the literature perform training on shorter sequences than the ones available in the training dataset. More specifically, it has been demonstrated that only a pair of neighboring frames and their annotations suffices for the model to learn temporal consistency. For example (Meinhardt et al. 2022) have shown that during pretraining, not even a tracking dataset is necessary, as neighboring frames can be obtained just by applying augmentations to an input frame. However, during finetuning neighboring frames are sampled from the tracking data sequences in MOT17 (Milan et al. 2016) and MOT20 (Dendorfer et al. 2020). We follow the last strategy and sample pairs of

images instead of longer sequences from the original 10-frame sequences, since this choice results in more data sequences to use as data samples during training and is sufficient for our model to learn temporal consistency.

The input images are not augmented with any standard image augmentation technique, since adding random noise or flipping increases the differences between neighboring frames to a level that hinders the model from recognising the same object in different sequences. However, scaling, rotations and occlusions are already naturally introduced in scenes during dataset generation from the simulated camera movement. Therefore, the only processing taking place with regard to the input images is a standard ImageNet-based normalisation. Though the images are not augmented, augmentation on a track level is performed during training by randomly sampling neighboring frames within the whole sequence length of the 10-frame sequences.

(a) Sequence with slow camera movement, all samples remain visible in every scene



(b) Sequence with faster camera movement that leads to occlusions

Figure 4.1.: Training dataset overview

4.2. Training setup

For the training of the proposed model the setup described in Tab. 4.1 is followed. Specifically, the dataset is loaded with a batch size of 2 and learning takes place with an AdamW optimizer. The learning rate is constant throughout the training and regularisation is ensured with a weight decay coefficient of 10^{-2}. The number of images the loss is computed on (forward images) is 1 and the horizon is also set to 1, meaning that for each forward image also 1 neighboring frame is loaded during dataloading to learn temporal consistency. This short sequence length during training (total sequence length = 2 frames) has been proven to be effective in training trackers in numerous works like (Meinhardt et al. 2022) and (Kipf et al. 2022). Finally, the maximum number of tracked objects is restricted to 5, which means that in scenes with more object annotations, some objects are treated as part of the background. This setup choice helps ensure that the network learns to track only designated objects and not every object present in the scene, potentially preventing overfitting.

Table 4.1.: Training setup

Parameter	**Configuration**
batch size	2
optimizer	AdamW
lr	10^{-5}
weight decay	10^{-2}
num of fwd ims	1
min/max horizon	1
max classes	5

4.3. Evaluation datasets

4.3.1. Evaluation Test set

For the method evaluation, a test set with longer sequences (minimum 50 frames) was generated with BlenderProc (Denninger et al. 2019). The test set generation process resembles the training set generation with regard to the scene characteristics. More specifically, a random number of objects (5-12) and a random table were sampled from the database to form the scene. However, as already explained in section 4.1, from the nature of scene generation with BlenderProc that involves simulating objects falling on a table and forming a pile, some objects may end up completely invisible for the whole scene. Hence, scenes with less that 5 visible objects are also part of the test set. In the end, an equal number of crowded ($num\_of\_objs < 6$) and not crowded ($num\_of\_objs >= 6$) scenes were sampled to include in the test set.

Table 4.2.: Test set sequences overview

	Sequence names	Num of frames	Num of objects	Visibility score	Type of Movement	Velocity
0	scene_0_zoom_in_slow	100	5	0.99	zoom_in	slow
1	scene_0_rotation_slow	100	5	0.97	rotation	slow
2	scene_1_zoom_in_slow	100	6	0.99	zoom_in	slow
3	scene_1_rotation_slow	100	6	1.00	rotation	slow
4	scene_2_zoom_in_slow	100	7	0.76	zoom_in	slow
5	scene_2_rotation_slow	100	6	0.70	rotation	slow
6	scene_3_zoom_in_slow	100	5	0.57	zoom_in	slow
7	scene_3_rotation_slow	100	5	0.70	rotation	slow
8	scene_4_zoom_in_slow	100	10	0.77	zoom_in	slow
9	scene_4_rotation_slow	100	10	0.78	rotation	slow
10	scene_5_zoom_in_slow	100	3	1.00	zoom_in	slow
11	scene_5_rotation_slow	100	3	1.00	rotation	slow
12	scene_6_zoom_in_slow	100	2	0.87	zoom_in	slow
13	scene_6_rotation_slow	100	2	0.94	rotation	slow
14	scene_7_zoom_in_slow	100	8	1.00	zoom_in	slow
15	scene_7_rotation_slow	100	8	1.00	rotation	slow
16	scene_8_zoom_in_slow	100	9	0.95	zoom_in	slow
17	scene_8_rotation_slow	100	9	0.95	rotation	slow
18	scene_9_zoom_in_slow	100	8	0.67	zoom_in	slow
19	scene_9_rotation_slow	100	8	0.84	rotation	slow
20	scene_0_zoom_in_fast	50	5	0.98	zoom_in	fast
21	scene_0_rotation_fast	50	5	0.97	rotation	fast
22	scene_1_zoom_in_fast	50	6	0.99	zoom_in	fast
23	scene_1_rotation_fast	50	6	0.99	rotation	fast
24	scene_2_zoom_in_fast	50	7	0.76	zoom_in	fast
25	scene_2_rotation_fast	50	7	0.74	rotation	fast
26	scene_3_zoom_in_fast	50	5	0.55	zoom_in	fast
27	scene_3_rotation_fast	50	5	0.70	rotation	fast
28	scene_4_zoom_in_fast	50	10	0.77	zoom_in	fast
29	scene_4_rotation_fast	50	10	0.78	rotation	fast
30	scene_5_zoom_in_fast	50	3	1.00	zoom_in	fast
31	scene_5_rotation_fast	50	3	1.00	rotation	fast
32	scene_6_zoom_in_fast	50	2	0.86	zoom_in	fast
33	scene_6_rotation_fast	50	2	0.92	rotation	fast
34	scene_7_zoom_in_fast	50	8	1.00	zoom_in	fast
35	scene_7_rotation_fast	50	8	1.00	rotation	fast
36	scene_8_zoom_in_fast	50	9	0.95	zoom_in	fast
37	scene_8_rotation_fast	50	9	0.95	rotation	fast
38	scene_9_zoom_in_fast	50	8	0.68	zoom_in	fast
39	scene_9_rotation_fast	50	8	0.83	rotation	fast

The generated test set involves 10 scenes with distinct number of objects and setup (table, background, lighting). In order to assess the robustness of different proposed model variants to occlusions and changes in object appearance caused by the camera movement, for each generated scene involving the same objects in the same environment, different camera movement types and camera velocities were sampled to form the final test set.

For each generated scene, two types of camera movements were sampled. The first type of camera movement encountered in the test set is primarily a translation movement towards the objects, simulating the case where the camera is approaching them. This type of translational movement is also encountered in case the robot is still, but the objects are moving towards it for example on a conveyor belt. In this scenario, the objects are observed from almost the same angle, so they appear to be scaled along the frames. The second type of movement is a rotational movement around the objects, simulating the case where the robot has approached its targets and is inspecting the scene to plan its movement for a manipulation task, e.g. grasping. This type of camera movement results in more dramatic appearance changes for the objects from scene to scene, since they are observed from different viewpoints and may become occluded along the course of the motion. As a result, the rotational category of movement sequences are more challenging to solve, though the type of movement is only one factor of difficulty in this test set.

Apart from the two different types of movements (zooming in/ rotational) in the same scene, two different camera velocities where sampled within the same scene for the sake of comparison. For that purpose, for each camera movement in each scene 200 frames where saved. For the scenes with the fast camera movement, the original registered 200 frames were sampled every 4 frames. For the slow movement variant, a subset of 100 frames from the original sequence was used. An overview of all generated scenes in all their variants in terms of camera movement type and velocity is provided in Tab. 4.2.

Aside from the movement type, the number of objects present in each scene and their visibility are involved in the analysis in chapter 5. The object visibility was estimated on a scene level by first taking the average over each class along all frames and then averaging over all the classes present in the scene according to:

$$vis_{score} = \frac{1}{|O_S|} \sum_{o \in O_S} \frac{1}{|F_{S(O)}|} \sum_{f \in F_{s(o)}} \frac{|o_{vis}|^f}{|o_{orig}|^f} \tag{4.1}$$

.

The visibility score is only computed on objects $|O_S|$ at least partly present in some frames of the scene and not on fully hidden objects in the whole sequence. The visibility score on a class level was estimated based on the ratio of the number of pixels of the visible part of the mask $|o_{vis}|$ with the total object mask $|o_{orig}|$. Naturally, the metric can be computed also on a single frame level in the sequence and in that case $|F_{S(O)}| = 1$.

4.3.2. Cross domain evaluation Test set DAVIS

Besides the synthetic test set generated with BlenderProc that shares the same characteristics with the training set, a standard semi-supervised tracking evaluation set, DAVIS (Pont-Tuset et al. 2017), was involved in the evaluation. Evaluating different versions of the proposed method on DAVIS serves as a cross-domain evaluation and is indicative of how well the method can predict unknown objects even from a different domain.

The DAVIS evaluation set consists of 82 sequences of different length. In practice only a subset of 30 sequences is used as an evaluation set after the train/val split. The sequences involve 1 to 5 classes, far less than the number of objects our model was trained on, and their length varies from 50 to 103 frames. The object annotations are also typically larger than the objects encountered in the training set scenes, as they may be animals or humans instead of everyday objects. Because of these differences with our training setup, performance on DAVIS could be seen as an indication of the models ability to generalise.

4.3.3. Stereo evaluation sequences shot with rc_visard 65

Additional to the two aforementioned test sets that include annotations and are suitable for a quantitative analysis, a small real life video test set was prepared for a qualitative assessment of the sim2real performance in the context of unknown object tracking of the proposed method. The sequences were shot in the lab of the Institute for Robotic and Mechatronic of the German Aerospace Center with the stereo RGB camera rc_visard 65 (Roboception 2023).

Two scenes with different characteristics that highlight different aspects of the proposed method are included in the evaluation:

- A relatively easy scene with 5 objects scattered on the floor and smooth camera movement to and away from the objects
- A more challenging scene with object occlusions and a more cluttered background on the conveyor belt of the lab

Since no annotations are available for the scenes, methods can be evaluated only qualitatively with regard to their performance. However, even this qualitative evaluation serves as a starting point for discussing advantages and limitations of the proposed method, especially in comparison with the single shot segmentation method INSTR (Durner et al. 2021).

4.4. Evaluation metrics

4.4.1. Tracking evaluation metrics

As already mentioned in the related work section, there have been a systematic attempt within the dynamic vision community to publish large densely annotated video datasets and establish challenges to standardise the comparison between tracking algorithms. Each

challenge is introduced along with its standard evaluation metrics, depending on the task formulation and the dataset characteristics.

For example, the standard tracking metrics used in the Multi Object Tracking (MOT) challenges are MOTA and IDF1. MOTA is a classic tracking metric first introduced by Stiefelhagen et al. 2007 and owes its popularity to its expressiveness, as it combines three sources of possible errors (False Negative (FN), False Positive (FP), ID switches (IDSW)) for the predicted tracklets. MOTA and its variants MOTSA and sMOTSA is already indicative of id consistency within a sequence by penalising id switches. IDF1 is a metric focusing primarily on track identity preservation capability over the entire sequence. Furthermore, higher order metrics that combine the detection and association quality assessment like HOTA (Luiten et al. 2020) and STQ (Weber et al. 2021) are in broad use by the community. For more details about tracking metrics, we refer the reader to the Appendix.

Despite the popularity of the MOT challenges metrics, Video Object Segmentation (VOS) challenges like DAVIS (Pont-Tuset et al. 2017) and YouTube-VOS (Xu et al. 2018) propose a different type of evaluation metrics that highlight better the shortcomings of methods typically proposed for the VOS task. More specifically, the J and F metrics are preferred as standard evaluation metrics in the evaluation kit for DAVIS. J is the Jaccard index, an IoU based metric that quantifies region similarity between estimated and ground truth masks for each tracked object in a sequence. F is the contour accuracy, focusing solely on correct object contour prediction. In general, there's a linear dependence between J and F (Pont-Tuset et al. 2017), hence calculating their mean $J\&F$ is also a reasonable step and used as a summary metric combining information of both region similarity and contour quality of prediction in sequences.

As already mentioned in chapter 2, our class agnostic task is closer to the VOS problem formulation. Additionally, we expect the predicted id of the tracked objects to be consistent by default, due to the matching process between past and current frame encodings proceeding the decoding process. On the other hand, we expect to encounter problems primarily in the segmentation quality of the predicted masks for the tracked objects. Therefore, the use of standard VOS metrics like $J\&F$ appears to be more reasonable to highlight issues in the performance of different model variants, instead of metrics focusing on id switches, like MOTA.

$$J = \frac{1}{|O_S|} \sum_{o \in O_S} \frac{1}{|F_{S(O)}|} \sum_{f \in F_{s(o)}} J(m_o^f, g_o^f) \tag{4.2}$$

$$F = \frac{1}{|O_S|} \sum_{o \in O_S} \frac{1}{|F_{S(O)}|} \sum_{f \in F_{s(o)}} F(m_o^f, g_o^f) \tag{4.3}$$

$$J\&F = \frac{1}{2}[m(J,S) + m(F,S)] \tag{4.4}$$

The details on the J, F and $J\&F$ metric computation on a sequence level are provided in (4.2), (4.3) and (4.4). It is worth noting that the per sequence value for both J and F metrics is computed by first taking the mean on a class level (computing the mean J/F value for all frames that a class is predicted) and then across all classes in the sequence. Finally, the overall J and F values for the whole test set are obtained by taking the mean over the per sequence computed values.

4.4.2. Single shot semantic segmentation evaluation metrics

.

In case of single shot semantic segmentation on a video sequence, each video frame is treated as an independent data input. Moreover, there is no notion of unique object ids, as each object is predicted anew in each incoming video frame. Therefore, segmentation quality can be only assessed on a frame level. A common evaluation metric in unknown instance segmentation also used in (Durner et al. 2021) is $mIoU$ on a frame level computed according to the formula:

$$mIoU = \frac{1}{|O_S|} \sum_{o \in O_S} \frac{TP_o}{TP_o + FP_o + FN_o} \tag{4.5}$$

where $|O_S|$ denotes the number of objects of interest that ought to be segmented in the scene.

To get a notion of a model's single shot segmentation performance on a video, where all frames are treated as independent samples, a mean value over all frames can be computed as in (4.6), similar to the volumetric $mIoU$ proposed by Zhan et al. 2022:

$$volumetric\_mIoU = \frac{1}{|F_{S(O)}|} \sum_{f \in F_{S(O)}} \frac{1}{|O_S|} \sum_{o \in O_S} \frac{TP_o^f}{TP_o^f + FP_o^f + FN_o^f} \tag{4.6}$$

.

4.4.3. Time and GPU measurements

For the time and GPU allocation measurements, inference was performed with 10 warm up cycles on the same GPU, a NVIDIA RTX A6000 with 48 GB memory.

4.5. Evaluation baselines

In this section, interesting aspects of the semi-supervised tracker HODOR (Athar et al. 2022) and the single shot instance segmentation method INSTR, both involved in the analysis in chapter 5, are presented. The focus is posed on details relevant to the analysis performed in chapter 5.

4.5.1. HODOR

HODOR leverages mask annotations from previous frames to propagate past frame information, similar to the mask encoder variant proposed in this work. However, in HODOR the past frame binary masks are directly used on the current frame feature map instead of a past frame encodings. Moreover, this masking operation only takes place on the smallest resolution feature map (1/32) and the mean of the masked features is subsequently led to an additional self-attention block with $N = 8$ attention heads. Then, the self-attented masked feature maps are led to a 'masked' attention block where the past downsampled ground truth masks are scaled with a learnable parameter a and concatenated with the inner product of keys and queries incoming from the self-attention block for the attention computation according to the formula:

$$masked\_attention(Q, K, V) = softmax(\frac{K^T Q + aM}{\sqrt{C}}) \tag{4.7}$$

HODOR also features an elaborate decoding process. More specifically, the proposed decoding strategy involves a deformable convolutional layer, and a cross attention layer similar to DETR (Carion et al. 2020) which result in very refined mask predictions, but also increase the computational complexity for the model.

A notable difference between HODOR and our method is the way small objects are treated during the learning process. In order to avoid involving all zero masks in the masking process due to downsampling, annotations that are smaller than a predefined area value are excluded from training. In our method instead, small objects are not ignored.

HODOR is pretrained on COCO (Lin, Maire, et al. 2015), a large dataset of static images. COCO features dense annotations of everyday scenes, from household objects like the ones encountered in our synthetic training dataset, to annotations of people or animals like the ones encountered on DAVIS, to categories unique to this dataset. COCO is converted to a tracking dataset for the purpose of training via augmentations. Specifically, sequences of 3 frames are generated from single images with flips and geometric transformations.

4.5.2. INSTR

INSTR (Durner et al. 2021) is a transformer based model for unknown object instance segmentation on stereo input images. Similar to DETR (Carion et al. 2020), feature extraction is performed with CNN layers, specifically twin ResNet-50 layers for the stereo input pair. After feature extraction, correlation layers (Dosovitskiy, Fischer, et al. 2015) are used to extract stereoscopic information. The correlated feature maps are then led to a transformer encoder/decoder block for dense mask prediction. Disparity is also deduced with an additional auxilary decoder.

INSTR leverages the power of transformers and achieves remarkable performance on single shot segmentation. Therefore, INSTR was chosen to serve as single shot segmentation baseline for the qualitative analysis performed in chapter 5. Moreover, similar to any reliable segmentation network, INSTR can be easily converted to a basic heuristic-based tracker with IoU association like the baseline used in (Weber et al. 2021). This naive INSTR+IoU tracker serves as a baseline in our experimental section, to highlight the performance gap between heuristic based and learning based tracking.

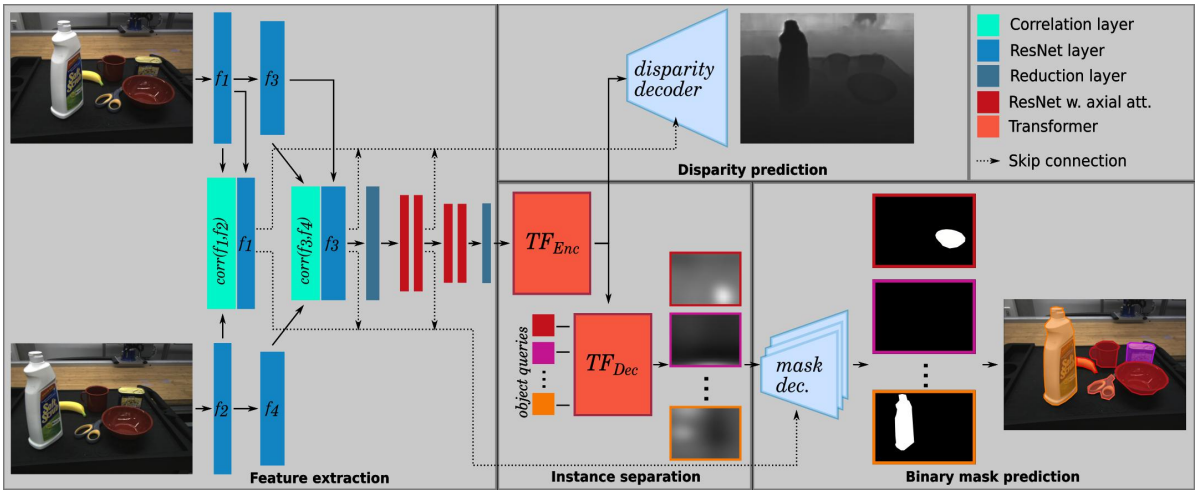


Figure 4.2.: INSTR model overview

The INSTR+IoU tracker utilises a matching mechanism with past frame predictions for successful association and track id assignment. More specifically, linear sum assignment with the Hungarian algorithm is performed between the previous and current frame predictions with a threshold for IoU overlap between predicted masks, typically provided as a hyperparameter. This matching mechanism between previous and current frame predicted binary masks results in track id propagation, essentially by reassigning the same track id to the most similar prediction in the current frame. An overview of IoU association matching is provided in Algorithm 2.

Algorithm 2 IoU association-based tracking

while $t < n_{frames}$ **do**
  $output_t = forward(inputs_t)$
  if t >0 **then**
    $matched\_output_t = hungarian\_matcher(output_t, past\_output)$
    $past\_output = matched\_output_t$
  else
    $past\_output = output_t$
  end if
end while

It is worth noting that the heuristic based tracker has many limitations and is expected to work well only in scenes with very slow camera movement that result in minimal appearance changes in objects between frames. Since track id propagation is performed based on IoU overlap with predictions from the past frame, any change in appearance that results in an IoU overlap <0.5 between past and current frame prediction results in unsuccessful matching and wrong track id assignment.

5. Results

The experiments involved in this chapter can be organised in two parts: First, a preliminary analysis on the differences between distinct model variants is presented. More specifically, different options for the past frame encoder, the matching strategy and the decoder are compared. Additionally, an expanded multi-scale version of the mask encoder with the option to include multiple descriptors per predicted class is involved in the comparisons.

Model variants are compared in terms of their performance on our test set as well as the cross domain evaluation set DAVIS. Apart from performance criteria, the inference time and GPU allocation are included in the method comparisons to indicate differences in the computational cost of each option.

In the second part of the analysis, the best performing variant of the proposed method is compared to a naive heuristic-based tracking baseline built on the unknown instance segmentation method INSTR and the semi-supervised tracking method HODOR. At the end of the chapter, a qualitative assessment of our proposed method with INSTR on recorded real-world data initialisation is presented.

5.1. Preliminary experiments

5.1.1. Encoder comparisons

The first type of performed model comparisons is between different types of past frame encoders. Specifically, the backbone image encoder, the bounding box encoder and the mask encoder are compared. The mask encoder is included in the analysis in two versions, with and without layernorm computation before matching, with the first option resembling more the masking strategy of HODOR (Athar et al. 2022).

All experiments are performed with the same attention-based matching mechanism and FPN decoder for a more fair comparison. The four encoder variants are compared in terms of their performance both on the synthetic tracking test set and the DAVIS evaluation set. It is worth noting that the DAVIS evaluation set consists of real images and provides an indication on the sim2real performance gap for models trained on synthetic data, that are meant to be deployed in real life scenarios. All methods are evaluated with regard to their performance on both datasets based on the achieved J, F, and$J\&F$ scores. For the tracking dataset, also the inference time is included in the evaluation Tab. 5.1.

Table 5.1.: Comparison of encoder types

Encoder Type	Tracking Dataset			Davis			time [s]
	J	F	J&F	J	F	J&F	
backbone	0.691	0.780	0.735	0.443	0.539	0.491	0.0986
bbox	0.600	0.703	0.651	0.361	0.442	0.406	0.0711
mask without mean	0.718	0.808	0.763	0.430	0.507	0.469	0.0704
mask with mean	0.708	0.797	0.752	0.438	0.524	0.481	0.0714

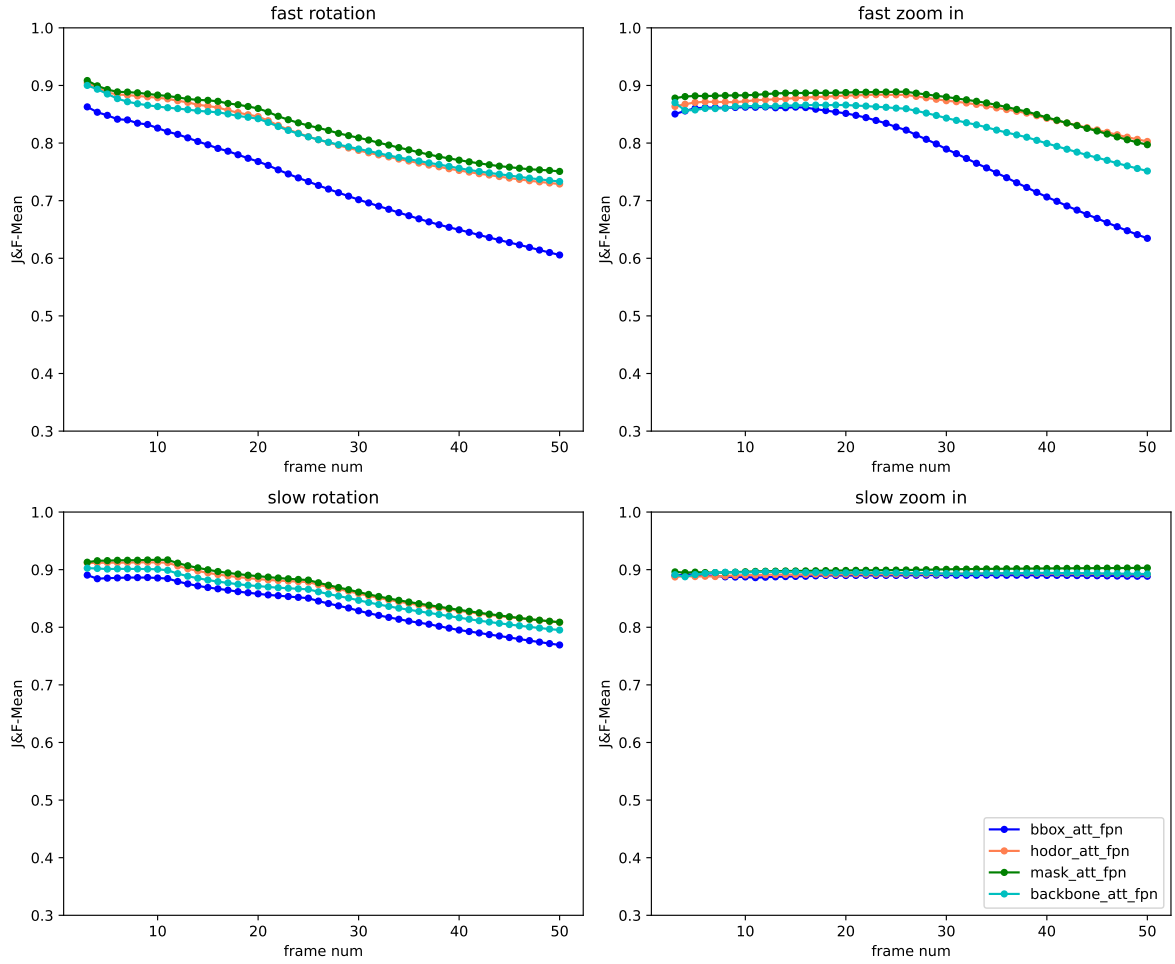


Figure 5.1.: Temporal analysis of encoder variants

Comparing the four variants in terms of their performance on the synthetic tracking dataset, the mask encoder without LayerNorm (denoted as 'without mean' in Tab. 5.1 before matching appears to achieve the highest *J&F* score. The backbone encoder also achieves comparable average performance, but requires longer inference time, since two twin SegFormer-B3 backbones are involved in the forward function. Finally, as expected, the bounding box

encoder achieves the poorest performance, since the past propagated information is merely a motion cue, instead of a rich past feature representation.

For a more detailed comparison of different encoder types, a temporal analysis was performed, where the cumulative $J\&F$ mean was computed for 50 frames on each scene. The test set scenes were divided into groups based on the type of camera motion encountered in the scene (rotational/zoom in) and the camera velocity (fast/slow). This grouping of the available scenes aims at better highlighting the differences in performance achieved by different encoding strategies depending on the scene difficulty, which is related to the camera movement type. Additionally, for a better indication of the range of performance scores between different scenes within each group, the variance is included in the plots, see Fig. 5.2.

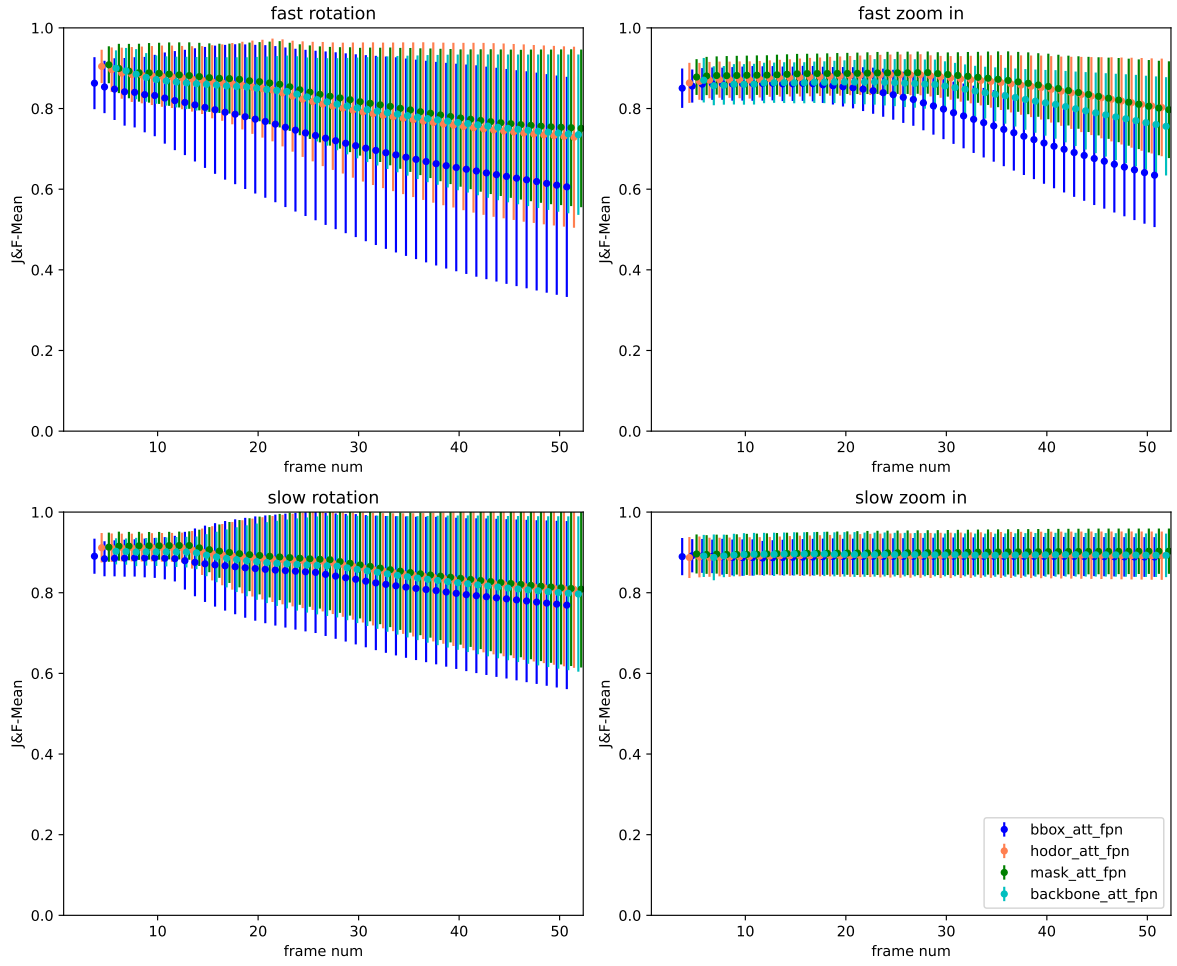


Figure 5.2.: Temporal analysis of encoder variants (with variance)

Observing Fig. 5.1, we note that in easier scenes, where the camera movement does not result in dramatic changes in object appearance (slow zoom in), all encoders perform comparably well in terms of the mean achieved $J\&F$ score, even when taking into account the performance

variance in Fig. 5.2. However, things appear to become critical in cases where the camera velocity increases. Both in case of a rotational and translational camera movement with fast velocity, the bounding box encoder performance drops dramatically as the sequence increases in length. It is worth noting that all methods' performance drops in fast scenes, probably due to accumulation of propagated errors over time, as the model performs inference on a long scene. Still, the bounding box encoder appears to be more affected in terms of performance as time progresses, which renders this model variant only suitable for cases where the movement is slow and controlled.

Corner case analysis

To better highlight the differences in behaviour between different encoder types, a 'crowdness/cluttereness' analysis was performed, to investigate the role of occlusions and cluttereness in the performance drop in difficult scenarios. For this analysis, the focus was placed on particularly difficult scenes, specifically the challenging 'fast rotation' sequence subgroup, based on the temporal behaviour of all encoder types summarized in Fig. 5.1. For the 10 sequences belonging to this subgroup, the *J&F* score, the visibility score, computed according to (4.1), and the number of objects present were plotted for 50 frames. The plots involving all 10 scenes can be found in section A.2 in the Appendix.

A more readable version of the plots in section A.2 is obtained by performing a corner case analysis. Specifically, for each past frame encoder type, the sequence with the largest and the smallest drop in performance indicated by the *J&F* mean was plotted along with the visibility score and the number of objects in the scene. To further ease the plot readability, the number of samples is only indicated in case it changes, otherwise it is only plotted with a rhombus in the first frame.

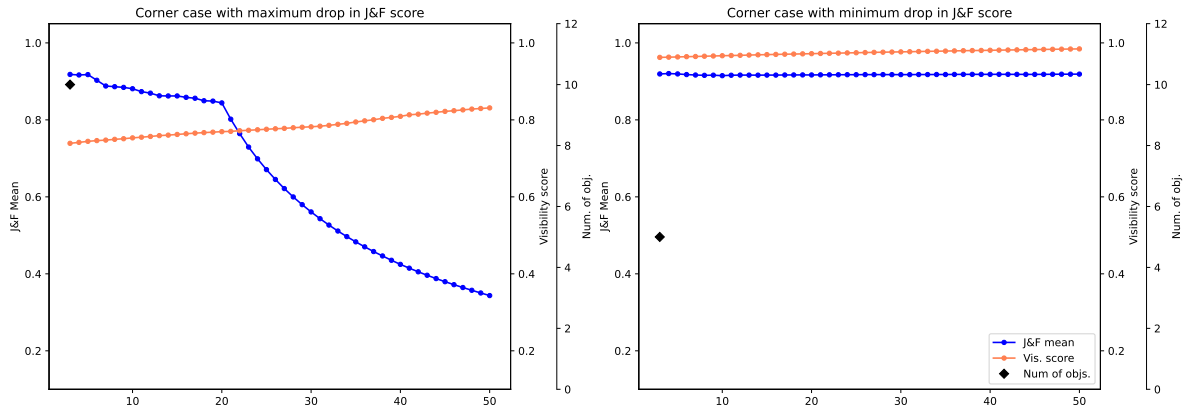


Figure 5.3.: Corner case analysis for the backbone encoder

Comparing the plots in Fig. 5.3, Fig. 5.4 and Fig. 5.5, it becomes evident that all methods struggle with very crowded scenes where a large number of objects is present. For the backbone

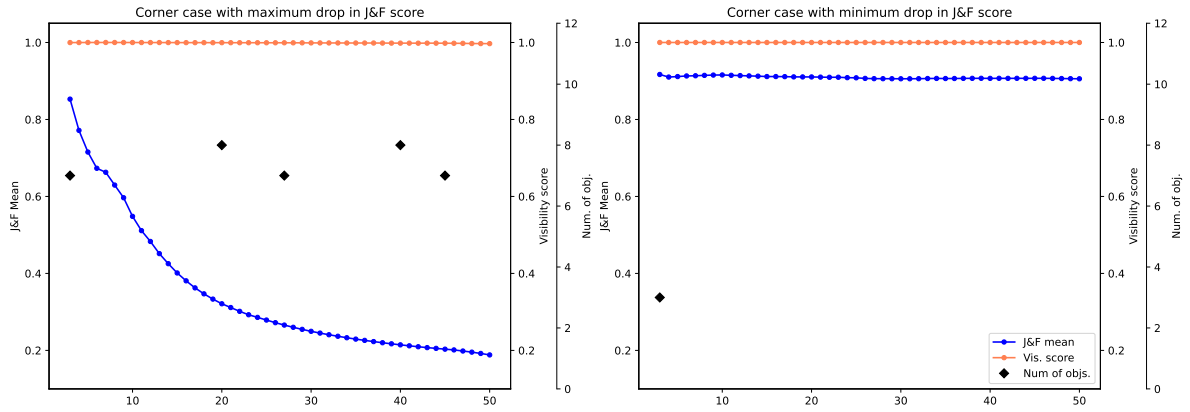


Figure 5.4.: Corner case analysis for the bounding box encoder

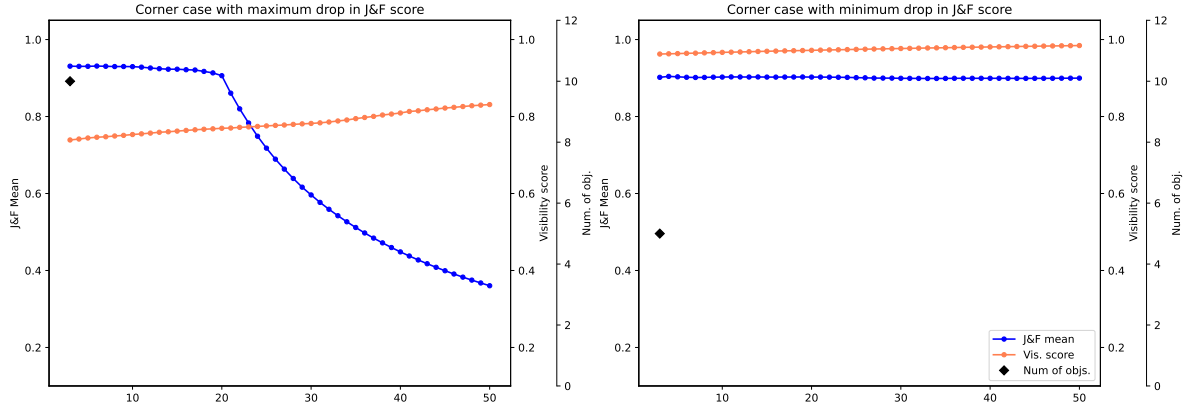


Figure 5.5.: Corner case analysis for the mask encoder

and the mask encoders, the most difficult scene involves a large number of objects (10 objects) combined with a low visibility score (<0.8), which implies that some of these objects are partially occluded by other objects in the scene, an indication of increased clutterness. On the contrary, both aforementioned encoders perform best in scenes with a relatively small number of objects (5 objects), no disappearances and a visibility score close to 1.0, which indicates that the objects are scattered far apart in the scene. Hence, both the backbone encoder and the mask encoders deal more easily with the appearance changes imposed to objects due to the camera movement, if the objects are far apart and easy to distinct. On the other hand, the bounding box encoder seems to struggle in a scene with an object disappearance, even though the rest of the objects are clearly visible and most probably far apart with a visibility score of 1. It also appears that the number of objects in the scene plays a more important role than how distinct the objects are in terms of their physical distance, as it performs best in a scene with only 2 objects.

Another important observation from the cluttereness analysis plots is that more sophisticated, appearance based methods (backbone encoder, mask encoder) seem to retain a relatively good performance for more frames ($\sim$ 20 frames), before the $J\&F$ score begins to drop after the 20th frame. In contrast, the performance drop of the bounding box encoder in its most challenging scene begins at t=1 and continues dramatically until the end of the scene.

5.1.2. Matching comparisons

For the matching type comparisons, three options were included in the analysis, all combined with the same encoder and decoder. More specifically, the mask encoder without mean was selected for past frame encoding, since it demonstrates best performance and retains best performance in the course of time even in challenging scenes (see Fig. 5.1). For the decoder, the standard FPN option was kept in all experiments in this subsection. The matching options involved in the analysis are matching by separable attention, e-step matching, which is a version GMM matching with only a single e-step of the EM algorithm performed, and GMM matching with 5 EM algorithm iterations. As mentioned in chapter 3, the number of iterations for the EM algorithm is defined as a hyperparameter.

Table 5.2.: Comparison of matching types

Matching	Tracking Dataset			Davis			time [s]	GPU	Lrn. Par.
	J	F	J&F	J	F	J&F			
attention	0.718	0.808	0.763	0.438	0.524	0.481	0.0709	1.255Gb	45.3M
estep	0.711	0.799	0.755	0.486	0.570	0.524	0.0727	423Mb	45.2M
GMM	0.702	0.802	0.752	0.416	0.468	0.439	0.0764	443Mb	45.2M

Taking into account solely the performance on our test set, all methods perform comparably well, even when considering the temporal aspect and the variance in the overall performance (see Fig. 5.6 and Fig. 5.7). On the cross domain evaluation set DAVIS, e-step matching seems to perform better than attention-based matching. This observation agrees with our intuition that GMM matching may result in a tracker less prone to overfitting to the specific training set it was optimised on, since the GMM matching block, unlike attention, does not involve learnable parameters. The role of GMM matching during training is merely to help optimise the input image encoder to extract features that are more suitable for identifying objects in different scenes. Hence, even though involving learnable parameters in the matching process evidently boosts performance in the domain the network is trained on, it is questionable whether the relation between features from different frames highlighted with attention is a similarity function as universal as the Gaussian Mixture assumption. Though not further examined in the course of this thesis, we think that an explainability analysis on the matching learned by the attention block in the proposed network could serve as interesting future work.

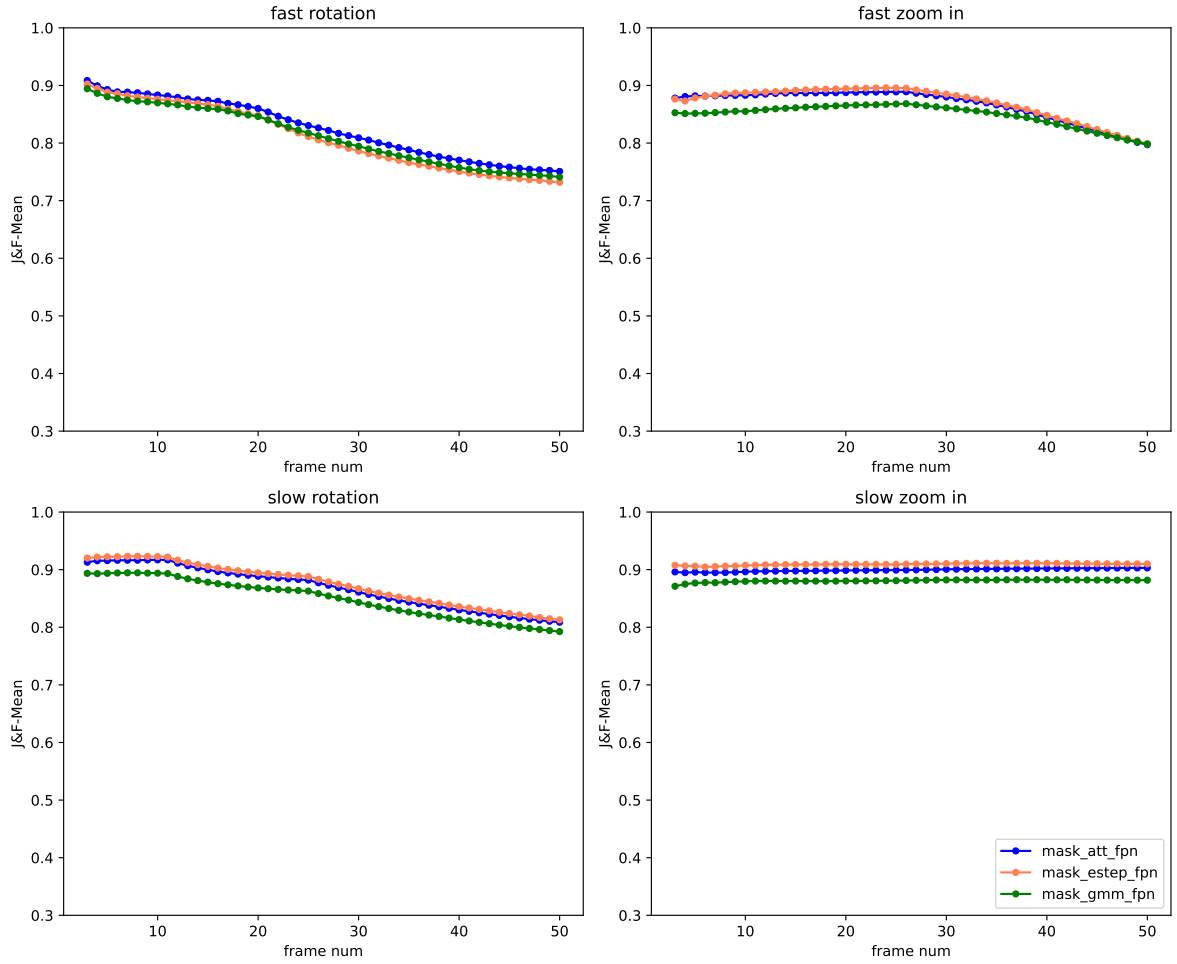


Figure 5.6.: Temporal analysis of matching type variants

Apart from purely performance-based criteria, an efficiency analysis was performed on all three options. In terms of inference time, the efficient separable attention proposed is faster than the other two options, probably due to the high parallelism of purely tensor-based operations involved in the method that run optimally with PyTorch. The GMM options on the other hand with the for loop introduced by the EM algorithm appears to require more time to execute. Since GMM matching is still a work in progress, the inference time is expected to decrease in a more mature version of the method and is left open as a research question. However, in terms of GPU allocation, the GMM variant is far less costly than attention. This difference in GPU allocation confirms the intuition described in chapter 3 that the GMM based matcher is more lightweight and hence more suitable for expanding in a more elaborate multi-scale matching variant. The expansion of GMM matching in its multi scale counterpart is explored in subsection 3.3.4.

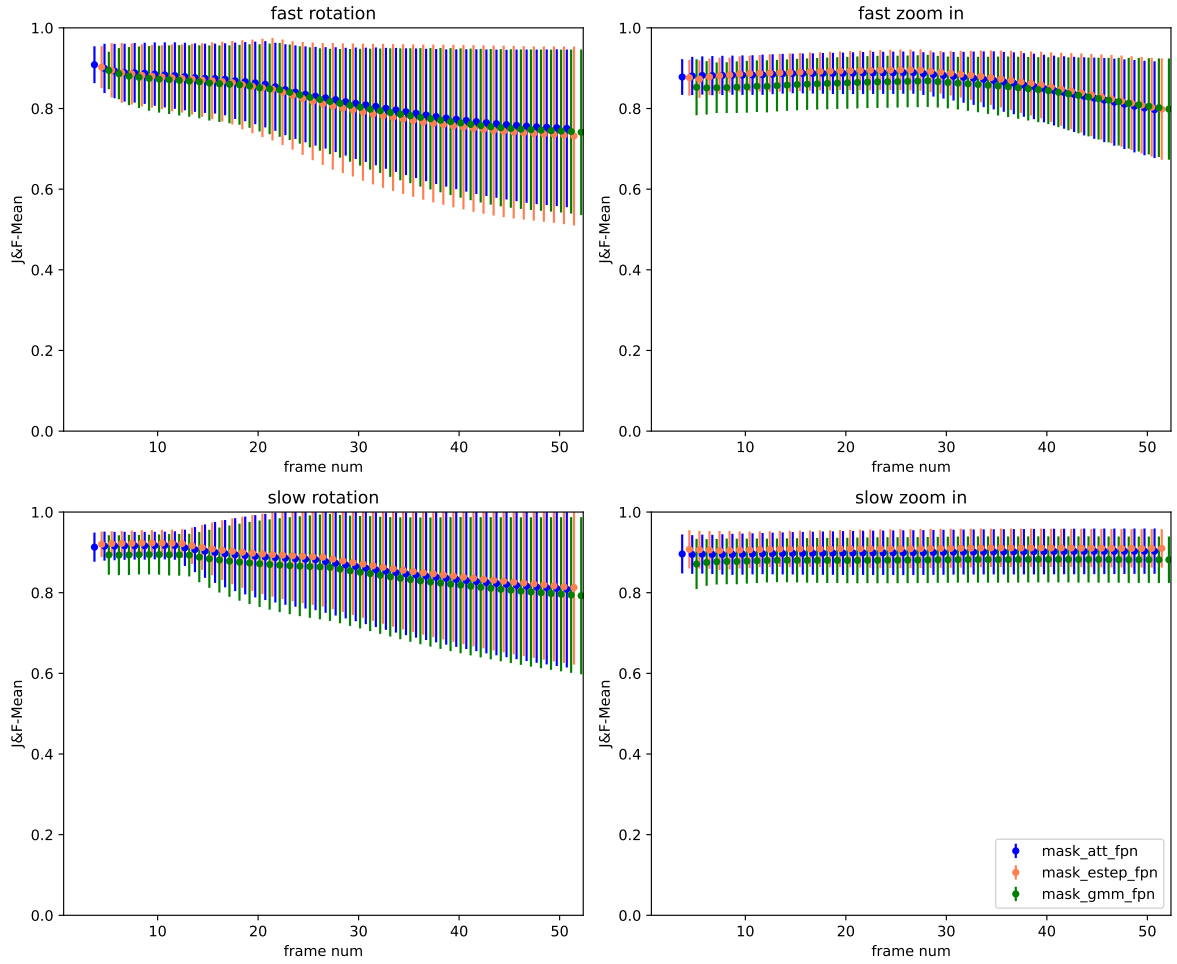


Figure 5.7.: Temporal analysis of matching type variants (with variance)

5.1.3. Decoder comparisons

Apart from the encoder and matching type comparisons, different decoders were tested with regard to their performance under the same setup. Similarly to the matching type comparison, the mask encoder without mean was used for past feature encoding in all experiments in this subsection. Regarding matching, the GMM-based matching was utilised in all compared model variants, because it is the only option compatible with all decoders discussed in this work, including the responsibility upsampler. Under this setup, the FPN decoder, the MLP decoder and the responsibility upsampler are compared primarily in view of their performance, but also in terms of their inference time.

Starting the analysis on our own synthetic dataset, we note that there is an important performance gap between the FPN/MLP solution and the responsibility upsampler. This result agrees with our expectations, since the responsibility upsampler directly decodes

Table 5.3.: Comparison of decoder types

Decoder Type	Tracking Dataset			Davis			time [s]
	J	F	J&F	J	F	J&F	
fpn	0.702	0.802	0.752	0.416	0.468	0.439	0.0764
mlp	0.707	0.781	0.744	0.373	0.488	0.43	0.0741
resp. decoder	0.624	0.677	0.650	0.364	0.384	0.374	0.1032

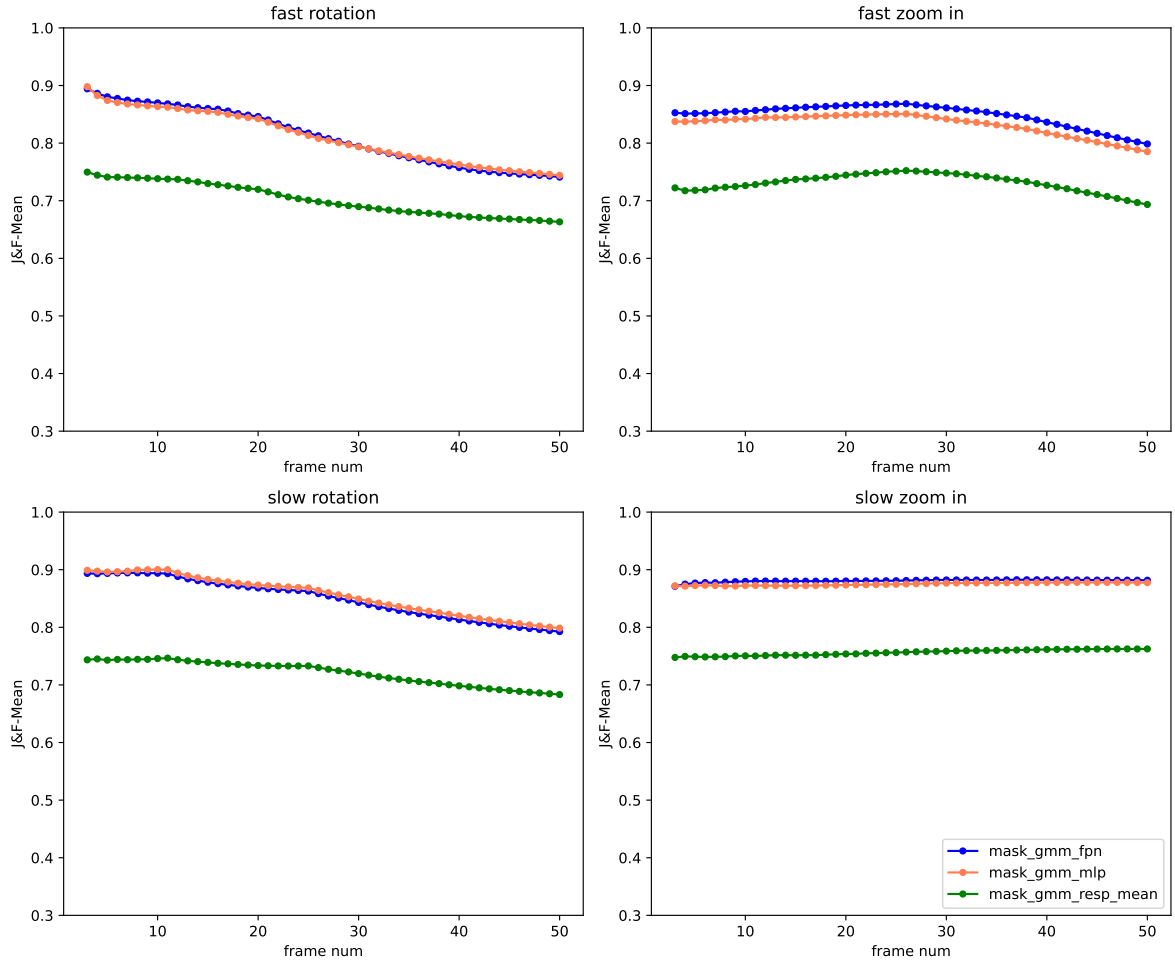


Figure 5.8.: Temporal analysis of decoder variants

the matched feature maps after the matching block and does not involve any more layers with learnable parameters that can perform some sort of filtering, that typically results in more refined predictions. The performance gap is evident in both the area based metric J, but also the contour focused metric F, in both our test set and the DAVIS evaluation, see Tab. 5.3. Moreover, even in terms of inference time the responsibility upsampler is more costly, since multiple GMM heads are required to achieve a relatively good performance.

So, as expected, the responsibility upsampler is not the optimal choice in view of solely performance or computational efficiency based criteria. However, as already mentioned in chapter 3, the responsibility upsampler provides more interpretable (due to their probabilistic nature) predictions, which may be a desirable aspect in various use cases.

Between the FPN and the SegFormer MLP-based decoder the performance gap is quite small. However, it seems that the pyramid multi-scale decoder architecture of the FPN still poses an advantage to the purely MLP based decoder from (Xie et al. 2021).

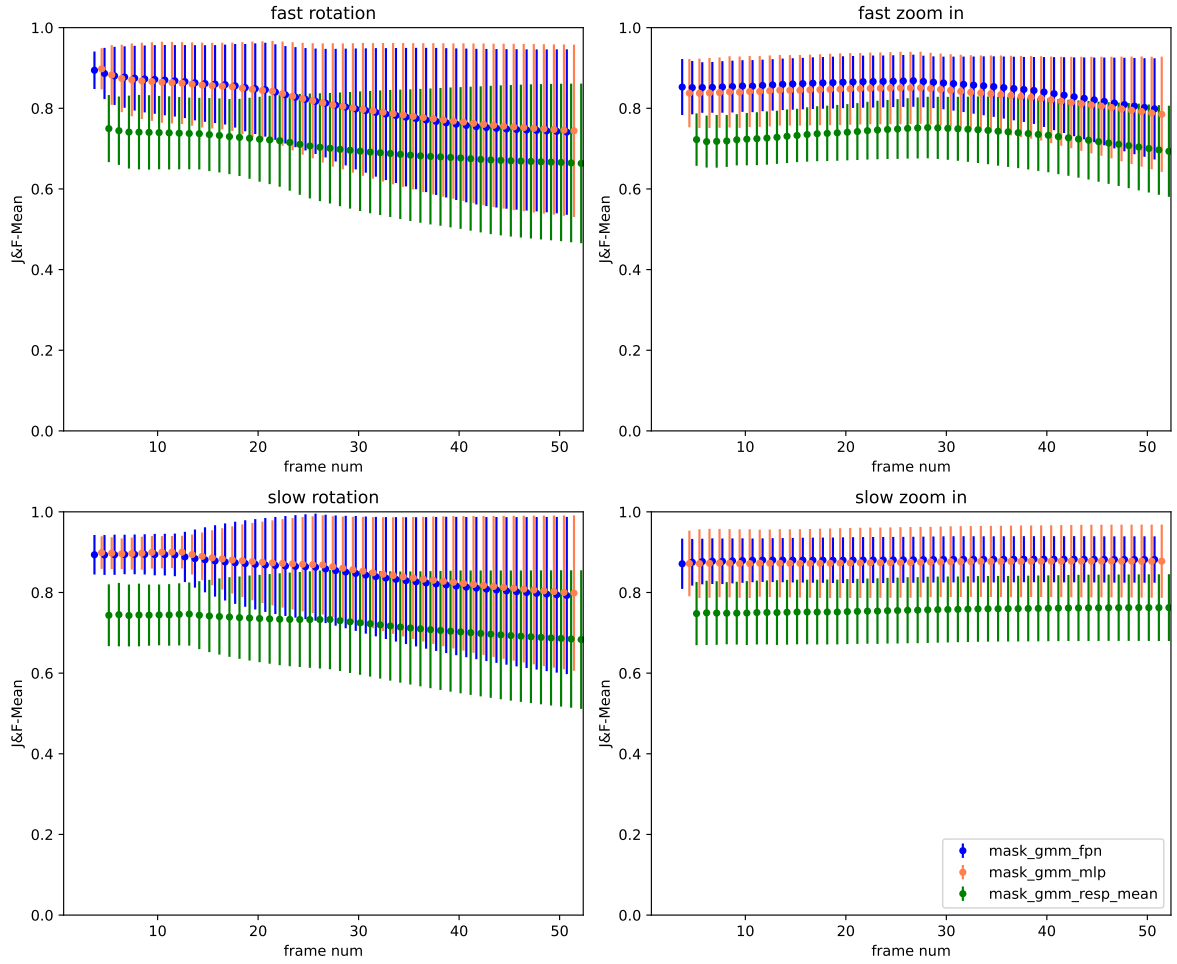


Figure 5.9.: Temporal analysis of decoder variants (with variance)

5.1.4. Comparison with multi-scale matching and multiple object descriptors

The final preliminary experiment described in this section is an attempt to expand GMM matching to a multi-scale variant. Both the backbone and mask encoder follow or have the potential to be expanded to a hierarchical structure since they build upon the SegFormer-B3 backbone for past feature encoding. However, matching so far has only been performed on the smallest feature map scale provided at the output of the encoder (1/32 of the original input image). Since hierarchical (Swin, Z. Liu et al. 2021, SegFormer, Xie et al. 2021) and pyramid structures (FPN, Lin, Dollár, et al. 2017) have proven to be beneficial for model performance, the question emerged whether multi-scale matching on all features in the respective resolutions can improve the performance of the proposed method.

Table 5.4.: Comparison of single-scale with multi-scale matching

Matching Type	Tracking Dataset			Davis			Time [s]
	J	F	J&F	J	F	J&F	
Mask enc. single-scale	0.702	0.802	0.752	0.416	0.468	0.439	0.0764
Mask enc. multi-scale	0.702	0.786	0.744	0.399	0.478	0.438	0.1030
Mask enc. multi-scale + descr	0.722	0.802	0.762	0.497	0.584	0.541	0.1340

Though expanding the patch encoder and the matcher to multiple feature scales seems intuitive, the computational cost of this operation immediately explodes, depending on the nature of the matching operation performed. For example, more elaborate architectures like HODOR (Athar et al. 2022) that involve self-attention on the past extracted features and perform a computationally intensive attention operation for matching (masked attention instead of our separable attention) cannot be expanded to a multi-scale variant, without a significant cost on computational resources. We considered more reasonable to expand the GMM matcher to multi-scale version, because it requires less GPU allocation compared to our separable attention and can be seen as more lightweight in that regard.

Apart from expanding matching to all extracted feature scales, a further attempt to boost performance would be to initialise multiple descriptors (clusters) for each object and the background class. More specifically, in this experiment 8 descriptors were initialised for the background class and 2 for each object. Especially, for the background class, we expect this choice to be very beneficial for performance, particularly in scenes where objects similar to the ones that are tracked are expected to be treated as background. By allowing the model to learn multiple descriptors for the background in scenes where objects are expect to be ignored, we expect the model to achieve better object of interest/ background separation. Our intuition is indeed verified by the preliminary results, see Tab. 5.4.

Observing the preliminary results on the our own test set, the multi-scale model variant with multiple background descriptors presents superior performance to all other model variants. Also in the temporal domain, we note that the multi-scale multi-background model retains a higher *J&F* score for more frames in the challenging fast zoom in group compared to the rest of model alternatives. Cross domain, on DAVIS, the multi-scale multi-background

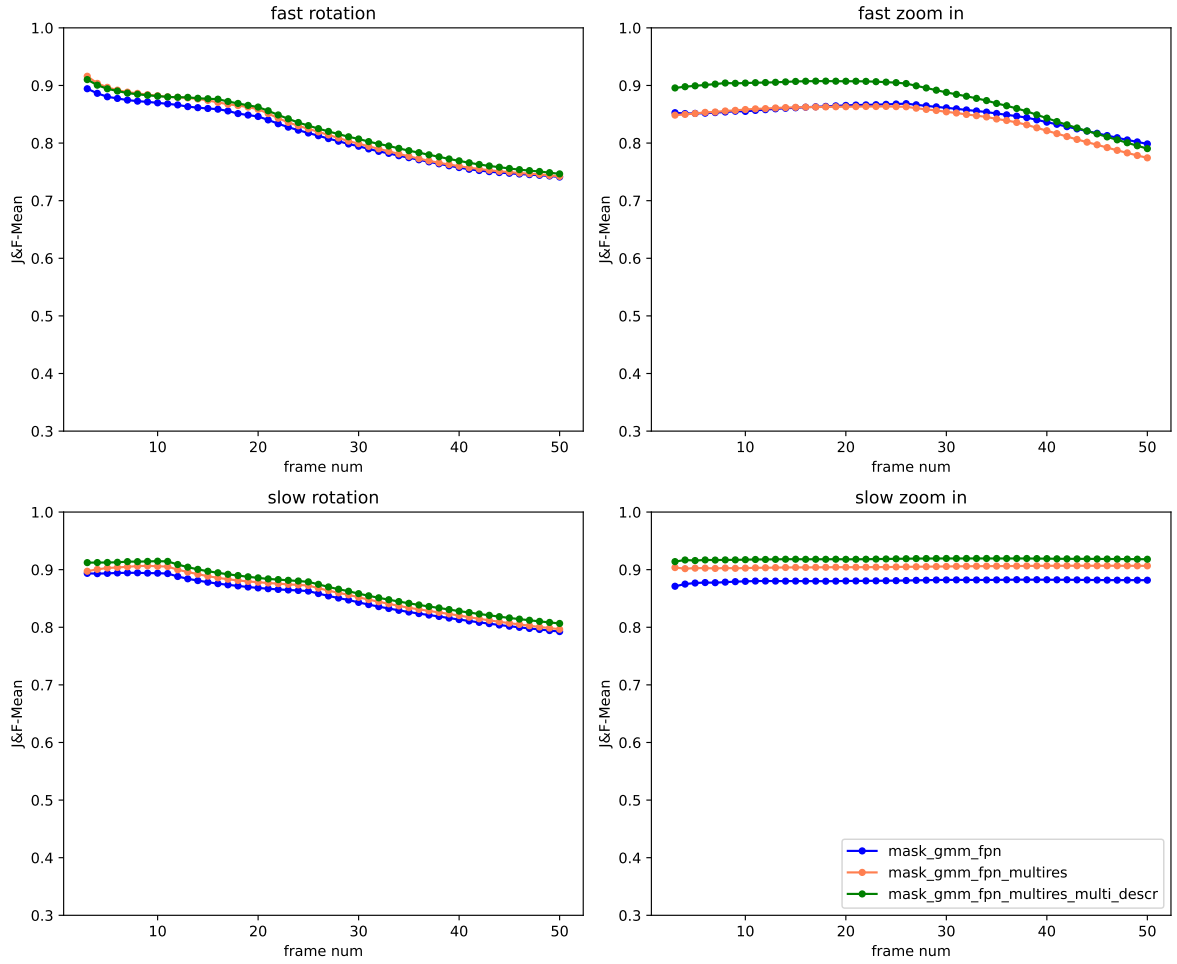


Figure 5.10.: Temporal analysis of single- vs multi-scale matching variants

model variant also presents a notable performance gap to the other options, which seems to confirm our hypothesis that multiple descriptors for the background result in more refined foreground/ background separation even in completely unknown setups. Naturally, the proposed multi scale multi background solution comes with the drawback of longer inference time, so in some cases it may not be preferred to other faster alternatives. However, as already mentioned, further optimisation in terms of resource allocation for the GMM matchers is possible, but is left as future work.

Another interesting observation from this analysis is related to multi-scale vs single-scale matching in the standard case with a single descriptor for both the background and the objects. From the data, it appears that the gain from just expanding the matching strategy to multi scale without multiple background descriptors is negligible, if not non existent. This finding agrees also with the little benefit observed when using the multi scale FPN decoder instead of the MLP decoder in the decoder section comparison.

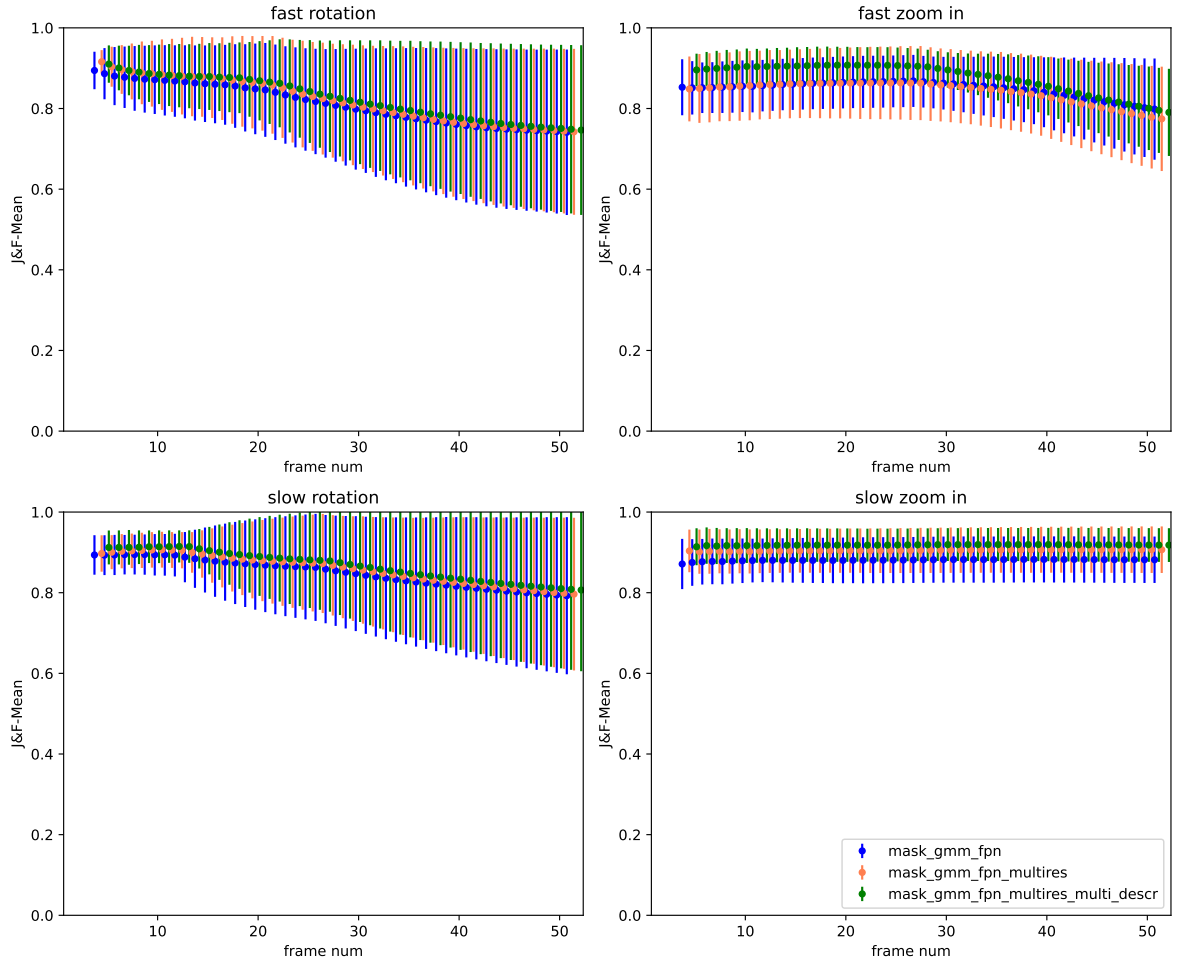


Figure 5.11.: Temporal analysis of single- vs multi-scale matching variants (with variance)

At the end of this section, comparing all possible model variants, the best performance is achieved with the multi scale multi background mask encoder, GMM matching and an FPN decoder. This model variant achieves a 0,762 $J\&F$ score on our test set and is used as our proposed method version, referred to as GMMTrack in the following section. Next, GMMTrack is compared with a baseline method and tracker. Furthermore, a qualitative analysis is performed on real life data, to provide an intuition about the magnitude of the sim2real gap resulting from training on a synthetic dataset but testing on real RGB stereo input data.

5.2. Final evaluation and experiments

5.2.1. Comparison with single shot segmentor

The first comparison of the proposed method with a baseline is with the single shot instance segmentation method INSTR (Durner et al. 2021). INSTR, like any single shot segmentation algorithm, can be converted to a naive heuristic tracker by adding a matching/association step with the previous frame prediction. In this work, INSTR is converted to a heuristic based tracker with IoU association. Details on INSTR+IoU are provided in subsection 4.5.2.

In general, the performance of association-based trackers is strongly dependent on the segmentation quality achieved by the neural network they build upon (Bewley et al. 2016). Hence, before converting INSTR into a baseline tracker, we ensure that the single shot segmentation performance on our test set achieves a high volumetric mIoU score, if all frames in all sequences are treated as individual test inputs. Indeed, INSTR achieves a 0,774 mIoU on our test set and is expected to perform adequately well as a tracker with IoU association.

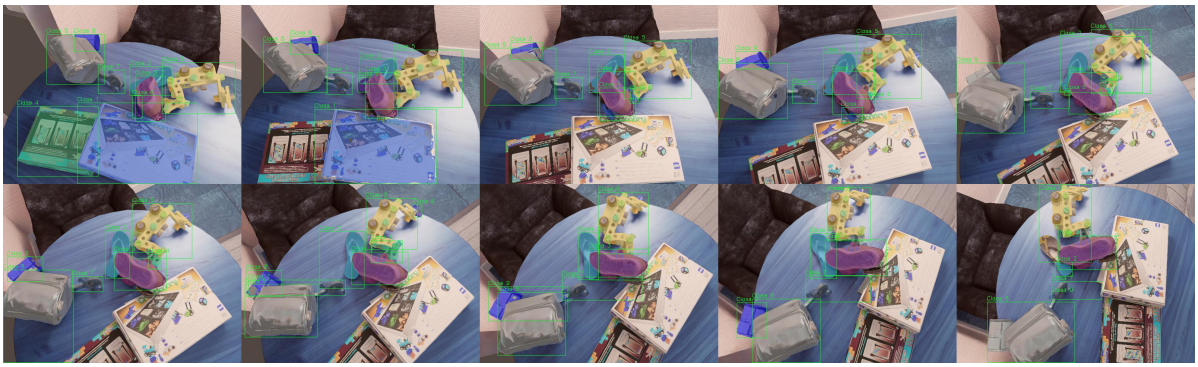
A qualitative overview of single shot segmentation with INSTR on a test set sequence, treating each frame individually, is provided in Fig. 5.12. For visualisation purposes, both in Fig. 5.12a and the ones that follow predictions are visualised every 5 frames in a 10 frame grid for a 50 frame sequence. For 100 frame long sequences, encountered in later experiments, predictions are included in the grid visualisation every 10 frames.

Table 5.5.: Comparison with single shot segmentation network INSTR

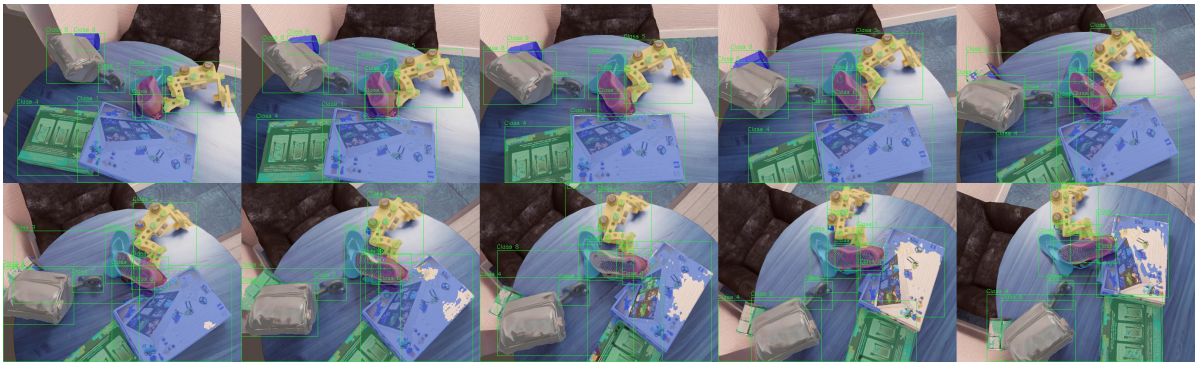
Model	vol. mIoU	J	F	J&F
INSTR	0.774	-	-	-
INSTR + IoU ass.	-	0.578	0.636	0.607
GMMTrack(ours)	-	0.722	0.802	0.762

After converting INSTR into a tracker with IoU association, we can assess the method on our tracking test set with the standard *J&F* tracking metrics used in the rest of our experiments. We note that the INSTR+IoU association achieves a 0,607 *J&F* score, which is lower than the worst performing variants of the proposed method, like the bounding box based encoder or the responsibilty upsampler with a mask encoder and GMM matching. However, this is expected, as heuristic-based trackers do not integrate past frame information to improve their predictions. The predictions result solely from the encodings of the current frame and are only assigned a temporally consistent track id by association with the previous frame prediction. If the network fails to detect an object in the current frame, due to occlusions or some other difficulty in the scene, the past frame encodings are not available to potentially guide the network to a more informed prediction, like in the case of learning-based trackers with some matching mechanism. Moreover, in case of dramatic appearance changes or non negligible displacement of an object between frames, the >0.5 IoU threshold is typically not satisfied and id assignment based on the IoU overlap does not result in correct id propagation for the same object in different frames.

(a) Inference on scene 4 rotation fast with INSTR



(b) Inference on scene 4 rotation fast with INSTR+IoU



(c) Inference on scene 4 rotation fast with GMMTrack

Figure 5.12.: Comparison with single shot segmentation network INSTR

A qualitative comparison between INSTR+IoU and GMMTrack in a difficult scene for INSTR+IoU is presented in Fig. 5.12b and Fig. 5.12c. As time progresses, IoU association fails for some classes and they are not included in the predicted binary masks. Common source of such prediction errors is oversegmentation or mask leaking in the estimated masks for the current frame. This faulty mask prediction results in failure of association with the predicted mask from the previous frame and initiates a spiral of error propagation in the sequence.

5.2.2. Comparison with SOTA

HODOR is compared to our method in two ways. Firstly, the pretrained model on static images is used out of the box on our test data. Secondly, the static images checkpoint is finetuned on our synthetic tracking dataset for 120000 iterations based on the provided recipe for training on sparsely annotated videos (Athar et al. 2020). Observing the achieved performance, it appears that finetuning on our dataset does not improve the performance obtained with the static image checkpoint. Evidently, the large scale training on COCO with image augmentations as past frames suffices in order for HODOR to achieve good performance on our test set and no new knowledge is added to the model via finetuning. To the contrary, the model seems to be led to overfitting, which is also reported by Athar et al. 2022 in some of their attempts to finetune on sparse video. In total, HODOR achieves similar performance to our proposed tracking method, probably due to the more sophisticated but computationally costly past information encoding mechanism with an additional self-attention mechanism and the elaborate masked attention on matching.

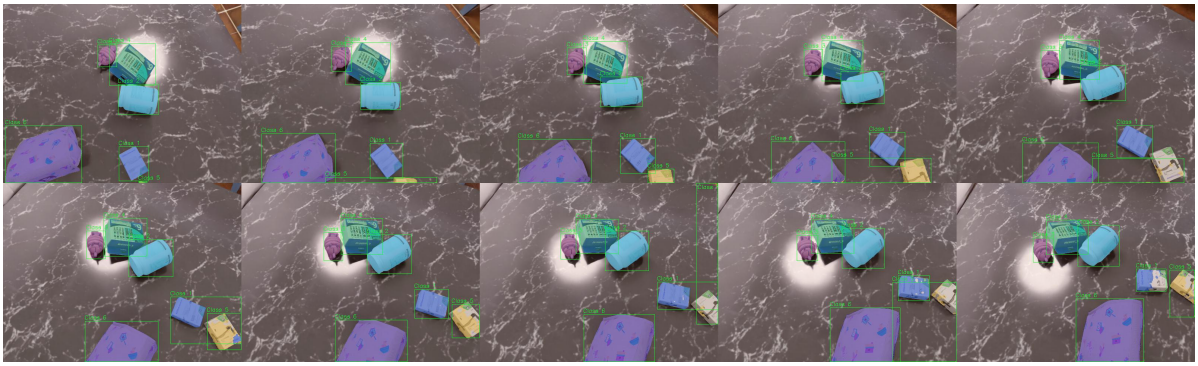
Table 5.6.: Comparison with SOTA semi-supervised tracker HODOR

Decoder Type	Tracking Dataset		
	J	F	J&F
GMMTrack (ours)	0.722	0.802	0.762
HODOR	0.720	0.790	0.755
HODOR finetuned	0.718	0.777	0.747

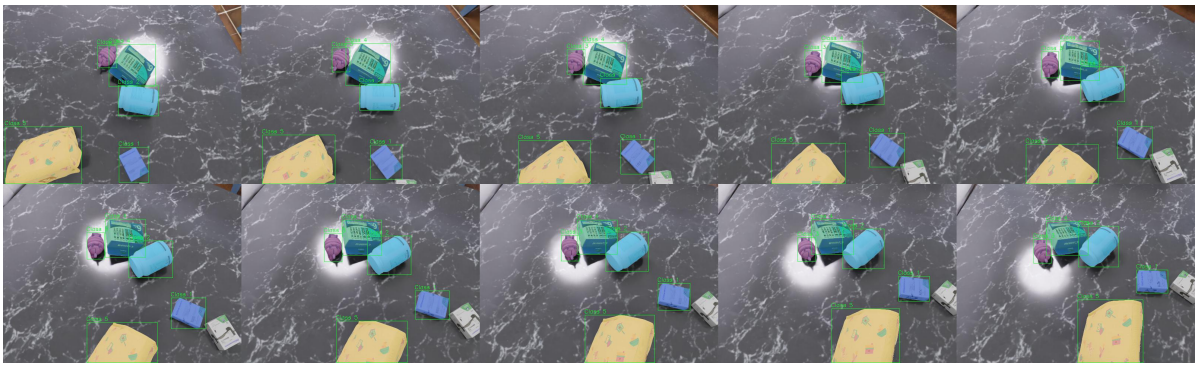
Qualitatively, HODOR predicts accurate masks both in terms of region similarity and contour accuracy, as can be seen in Fig. 5.13b. However, HODOR in all its variants completely ignores small objects, as in the case of class 5 in scene 1 of our synthetic dataset, see Fig. 5.13b. For comparison, the predictions with our model are visualised every 5 frames in Fig. 5.13a. It is evident that our method predicts all classes regardless of the annotation size, including class 5 with a very small annotation in frame 0. This behaviour of HODOR is expected, since small objects are filtered out during initialisation and excluded from training. This might be one of the reasons why even after fine tuning on our training set, HODOR does not improve much in performance. Indeed, regardless of the accuracy achieved on a region similarity or contour level, completely ignoring classes in the predictions is heavily penalised by the $J\&F$ metric.

5.2.3. Qualitative analysis on real data

Apart from the quantitative analysis presented in the previous sections, a qualitative analysis on real data was performed to provide an intuition on the sim2real gap for our method after training on a synthetic dataset. Ideally, this analysis would have taken place quantitatively. However, creating a real life annotated test set, with sequences recorded during real robotic manipulations, exceeds the scope of this thesis.



(a) Inference on scene 1 rotation slow with GMMTrack



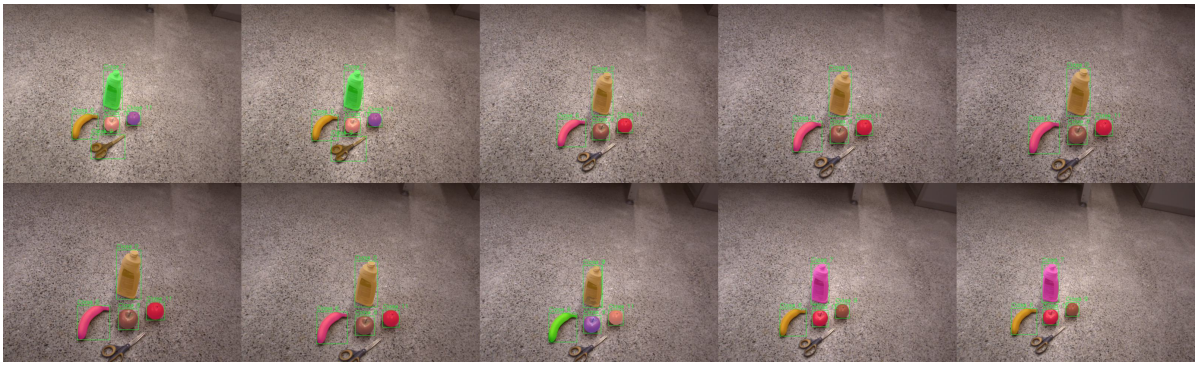
(b) Inference on scene 1 rotation slow with HODOR

Figure 5.13.: Comparison with semi-supervised tracker HODOR

Without having access to annotations, initialising our method is slightly more challenging. In practice, GMMTrack is initialised with predictions provided by the single shot segmentation network INSTR for the purpose of this analysis. In theory, this initialisation can also happen with a human in the loop in a real life scenario, and this is also an interesting direction left as future work.

The GMMTrack predictions with INSTR initialisation are compared with single shot INSTR predictions (without tracking) in two setups. In the first scene, the objects are scattered in front of a relatively neutral background, with distance between them. Even though some furniture parts are visible in the back of the frames, they are not included in the predicted segmentation masks, neither by GMMTrack see Fig. 5.14b nor by INSTR see Fig. 5.14a. The only difference between the two networks is that GMMTrack, as expected, predicts temporally consistent masks, since it assigns the same class for the same object in the whole sequence. This is not the case for INSTR, though in this easy scene, just adding the IoU association step would most probably be enough to correctly assign the ids throughout the frames, since the objects change only slightly in scale and appearance in the course of the sequence.

The second scene included in the analysis is shot on the conveyor belt of the robotic lab and is far more challenging than the floor scene. The camera movement is more abrupt and results

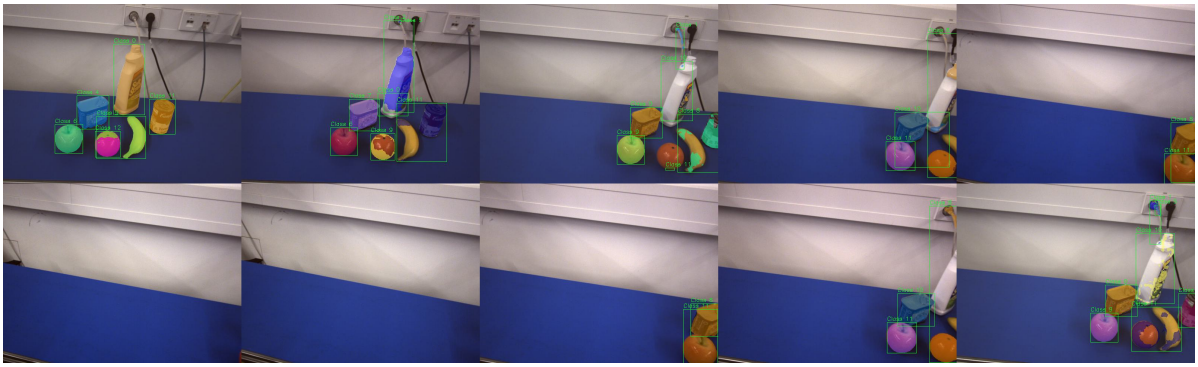


(a) Inference on floor sequence with INSTR

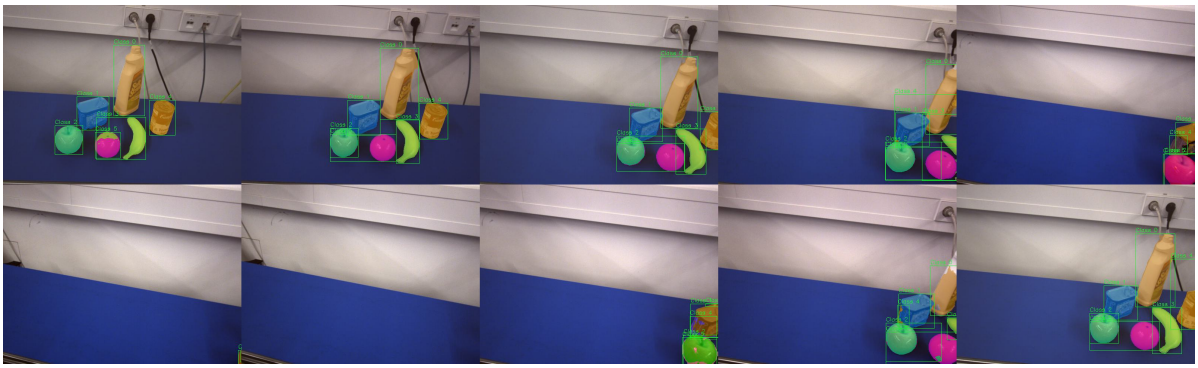
(b) Inference on floor sequence with GMMTrack

Figure 5.14.: Inference on floor sequence

in dramatic occlusions and reappearances of some objects. Additionally, more background clutter is present in the scene, as various sockets and cables from the lab equipment are visible. Here, it becomes evident how important correct initialisation is for GMMTrack's prediction quality on the sequence. If the network is initialised with correct segmentation masks like in Fig. 5.15b, the masks are correctly predicted for the first 25 frames (frames are included in the grid every 5 frames). Moreover, after the occluded objects reappear in frame 40, the correct ids are recovered. In comparison, INSTR appears to provide worse segmentation quality in cluttered scenes. By frame 15 in Fig. 5.15a the soft-scrub object is no longer detected. On the contrary sockets and background clutter are detected as objects of interest. This scenario seems to validate our original hypothesis that tracking can also help improve segmentation quality in challenging, real life, cluttered scenes.



(a) Inference on conveyor belt sequence with INSTR



(b) Inference on conveyor belt sequence with GMMTrack

Figure 5.15.: Inference on conveyor belt sequence

6. Conclusion

In the course of this thesis, a semi-supervised tracker for unknown objects with a novel past frame encoder and a matching module is proposed. Our method can be used to guide the actions of mobile robotic agents deployed in dynamic, real-world settings, that may involve cluttered scenes with randomly scattered small objects. For both the past frame encoder and the matching module, different options are examined. In this context, the following three research questions were formulated:

1. How do different options for past frame encoding and matching compare with respect to their performance and computational cost?
2. Does our method outperform semi-supervised trackers optimized for a different domain?
3. Can tracking of segmentation masks be beneficial for the overall achieved segmentation quality compared to single-shot prediction in challenging scenarios?

Firstly, we summarize our findings from chapter 5 regarding the novel, tracking-related modules in the proposed DL network. For the past frame encoder, three different options were investigated, from the very lightweight bounding box encoder to the computationally intensive backbone encoder and the powerful and lightweight mask encoder. We conclude that the mask encoder combined with any matching mechanism is more suitable for robotic tracking, as it performs better in more challenging scenes with object occlusions and dramatic changes in appearance. Nonetheless, the bounding box encoder combined with separable attention matching may also pose an interest in use cases where the environment is controlled and the camera movement is slow and purely translational, like in the case of a conveyor belt scene without much background clutter. Finally, the initial attempt to encode past frame information with a siamese backbone encoder similar to our input feature encoder performs slightly worse than the mask encoder, but adds a computational cost in the pipeline that makes arguing for its deployment harder.

Comparing the proposed matching mechanisms, separable attention achieves better performance than GMM matching in the single-scale setup with single object and background descriptors. However, GMM-based matching outperforms attention in its more elaborate multi-scale variant with multiple background/object descriptors. This finding agrees with our initial intuition that multiple object descriptors may be beneficial for effective foreground/background separation in cluttered scenes like these encountered in the robotic setup. Moreover, from the cross-domain analysis on DAVIS, GMM matching seems to be

more promising for sim2real deployment in real-world applications. Nonetheless, any conclusions regarding sim2real applicability require further quantitative analysis on a real-world dataset, which is left as future work. In conclusion, our GMM matching in its multi-scale, multi-descriptor variant outperforms attention-based matching in the experiments and can be considered the optimal option combined with the mask encoder for tracking unknown objects in the robotic setup.

Regarding the second research question, we summarize the results of the comparison between the semi-supervised tracker HODOR, designed for a different domain, and our proposed method, which is optimized for the particular attributes of the robotic setup. HODOR achieves an overall comparable performance to our best-performing method variant due to its very accurate mask predictions, both in terms of region similarity and contour accuracy of the predicted masks. However, we demonstrated that HODOR fails to track small objects and in that regard does not satisfy the requirements for an optimal robotic tracker as formulated in chapter 1.

Lastly, our third research question was related to whether segmentation quality can benefit from past feature propagation in challenging scenes. If temporal consistency is completely ignored and each input frame is treated as independent, at least in the synthetic test set, the single shot segmentation network INSTR performs comparably well with our proposed network on average. However, we demonstrated that in very cluttered scenes with abrupt camera movement, the predicted mask quality from our tracker outperforms the one from INSTR (see Fig. 5.12). Additionally, the qualitative results on the real-world scenes recorded in the lab imply that past frame feature propagation is beneficial for the robustness of predictions even in easier scenes (see Fig. 5.14).

To conclude, GMM matching poses an interesting alternative to attention-based matching in the context of unknown object tracking for robotic applications and opens up a promising direction for further future investigation. For example, involving more than one past frame in the matching process between current and past frames could be an interesting method enhancement. Different works in current literature like (Z. Yang, Wei, et al. 2021) and (H. K. Cheng et al. 2022) propose more elaborate memory modules to process and encode multiple past frames in order to improve their performance. Apart from matching with features sampled from multiple past features, we hypothesize that enriching the past frame features with encoded past frame predictions during training could further improve the robustness of the proposed object-tracking segmentation network.

A. Appendix

A.1. Tracking metrics overview

Table A.1.: Overview of popular tracking metrics

Metric	Datasets/ Challenges	Formula
MOTA	MOT Challenges	$MOTA = \frac{\lvert TP\rvert - \lvert FP\rvert - \lvert IDSWc\rvert}{\lvert GT\rvert}$
IDF1	MOT Challenges	$IDF1 = \frac{\lvert IDTP\rvert}{\lvert IDTP\rvert + 0.5\lvert IDFN\rvert + 0.5\lvert IDFP\rvert}$
HOTA	MOT Challenges	$DetA_\alpha = \frac{\lvert TP\rvert}{\lvert TP\rvert + \lvert FN\rvert + \lvert FP\rvert}$, $AssA_\alpha = \frac{1}{\lvert TP\rvert}\sum_{c\in TP} A(c)$ $A(c) = \frac{\lvert TPA(c)\rvert}{\lvert TPA(c)\rvert + \lvert FNA(c)\rvert + \lvert FPA(c)\rvert}$, $HOTA_\alpha = \sqrt{DetA_\alpha AssA_\alpha}$
MOTSA	MOTS Challenge	$MOTSA = \frac{\lvert TP\rvert - \lvert FP\rvert - \lvert IDSWc\rvert}{\lvert GT\rvert}$
sMOTSA	MOTS Challenge	$sMOTSA = \frac{\widetilde{TP} - \lvert FP\rvert - \lvert IDSWc\rvert}{\lvert GT\rvert}$ $\widetilde{TP} = \sum_{h\in TP} IoU(h, c(h))$
J	DAVIS/YouTube-VOS	$m(J,S) = \frac{1}{\lvert O_S\rvert}\sum_{o\in O_S} \frac{1}{\lvert F_{S(O)}\rvert}\sum_{f\in F_{s(o)}} J(m_o^f, g_o^f)$
F	DAVIS/YouTube-VOS	$m(F,S) = \frac{1}{\lvert O_S\rvert}\sum_{o\in O_S} \frac{1}{\lvert F_{S(O)}\rvert}\sum_{f\in F_{s(o)}} F(m_o^f, g_o^f)$
AP, AP50, AP75	YouTube-VIS	$IoU(i,j) = \frac{\sum_{t=1}^{T} \lvert m_t^i \cap m_t^j\rvert}{\sum_{t=1}^{T} \lvert m_t^i \cup m_t^j\rvert}$ $AP = mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$
STQ	MOTChallenge-STEP	$AQ = \frac{1}{\lvert gt_t racks\rvert}\sum_{g\in gt_t racks} \frac{1}{\lvert gt_{id}(g)\rvert}\sum_{p,\lvert p\cap g\rvert\neq\emptyset} TPA(p,g) x IoU_{id}(p,g)$ $SQ = mIoU = \frac{1}{\lvert C\rvert}\sum_{c\in C} \frac{\lvert pr_{sem}(c)\cap gt_{sem}(c)\rvert}{\lvert pr_{sem}(c)\cup gt_{sem}(c)\rvert}$ $STQ = (AQxSQ)^{\frac{1}{2}}$

A.2. Crowdness analysis encoders

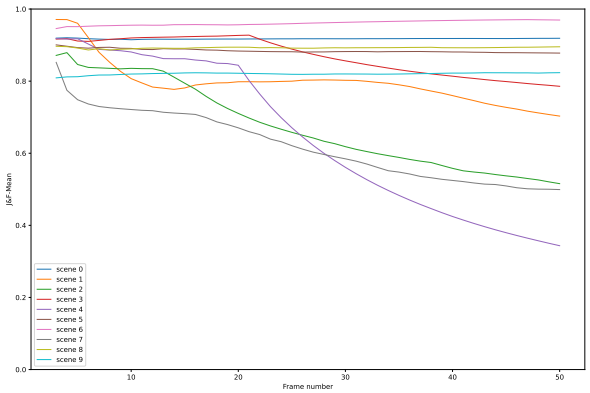


Figure A.1.: J&F-Mean backbone encoder on all rotation fast sequences

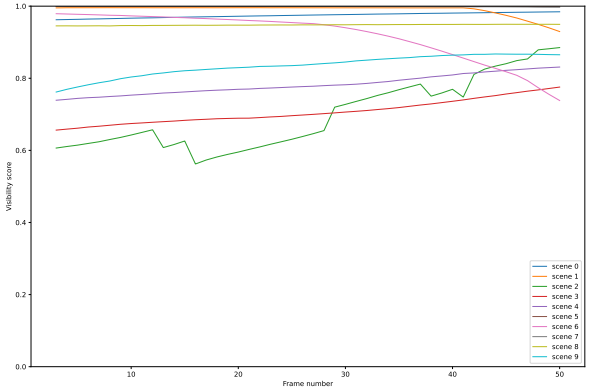


Figure A.2.: Visibility score backbone encoder on all rotation fast sequences

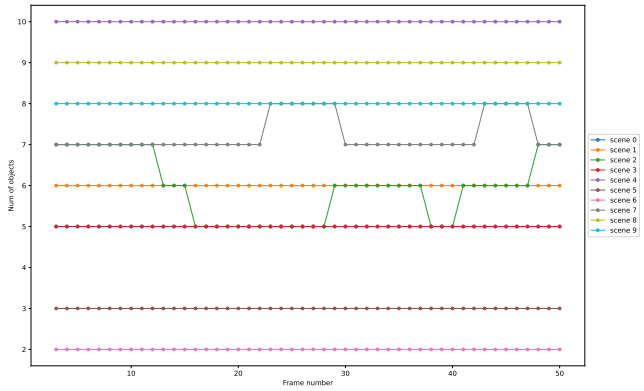


Figure A.3.: Num. of objects backbone encoder on all rotation fast sequences

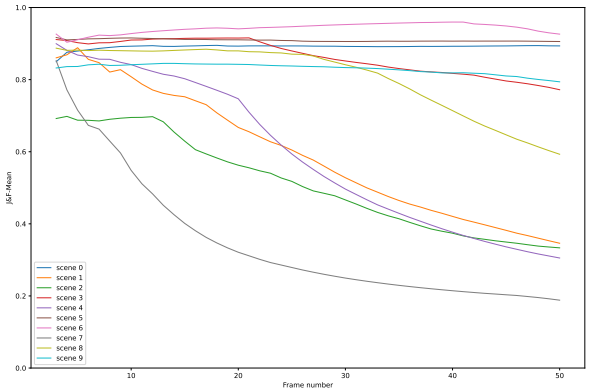


Figure A.4.: J&F-Mean bounding box encoder on all rotation fast sequences

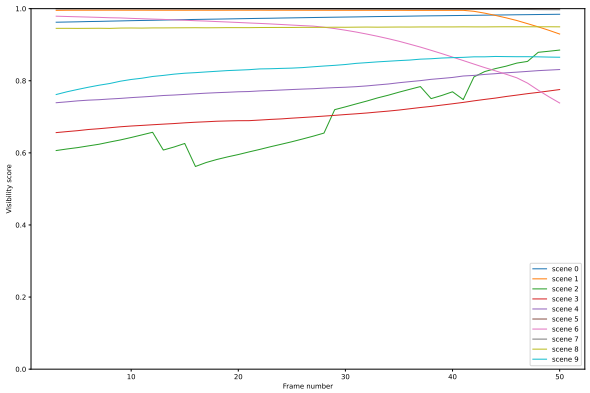


Figure A.5.: Visibility score bounding box encoder on all rotation fast sequences

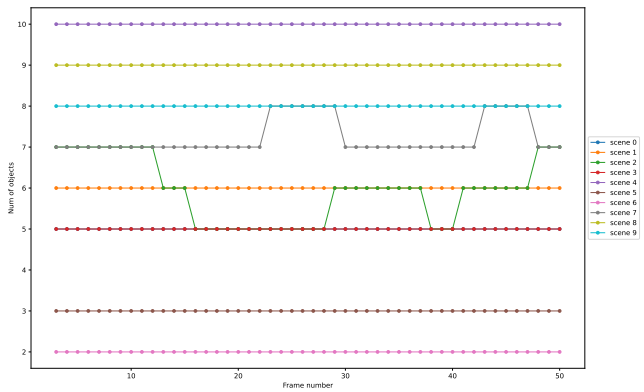


Figure A.6.: Num. of objects bounding box encoder on all rotation fast sequences

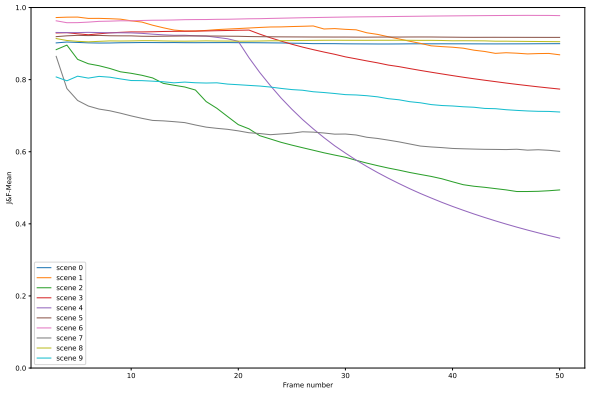


Figure A.7.: J&F-Mean mask encoder on all rotation fast sequences

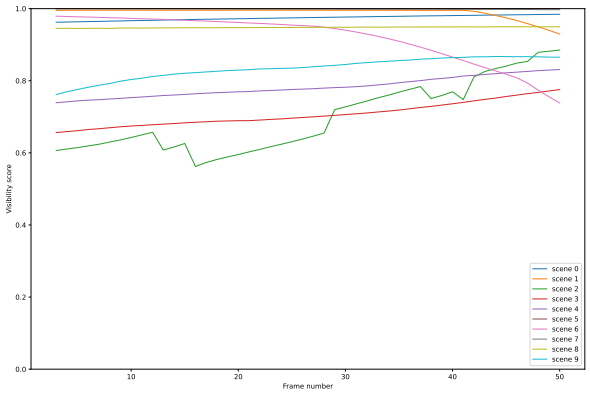


Figure A.8.: Visibility score mask encoder on all rotation fast sequences

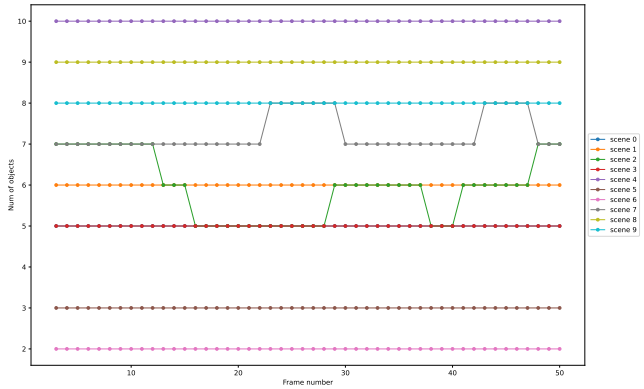


Figure A.9.: Num. of objects mask encoder on all rotation fast sequences

List of Figures

List of Tables

Glossary

BlenderProc Modular procedural pipeline, which helps in generating real looking images for the training of deep neural networks. 23

sim2real Knowledge transfer from simulation to real life problems. 34

Acronyms

BB bounding box. 14

DL Deep Learning. 2

e-step Expectation step. 17

FN False Negative. 29

FP False Positive. 29

GMM Gaussian Mixture Model. 15, 53

IDSW ID switches. 29

IoU Intersection over Union. 7

MOT Multi Object Tracking. 29

m-step Maximization step. 17

MLP Multilayer Perceptron. 12

MOTS Multi Object Tracking and Segmentation. 4

SOTA State-Of-The-Art. 6

VIS Video Instance Segmentation. 4

VOS Video Object Segmentation. 4, 29

VPS Video Panoptic Segmentation. 4

Bibliography

Sundermeyer, M., Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel (2018). "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images". In: *The European Conference on Computer Vision (ECCV)*.

Sundermeyer, M., A. Mousavian, R. Triebel, and D. Fox (2021). "Contact-GraspNet: Efficient 6-DoF Grasp Generation in Cluttered Scenes". In.

Boerdijk, W., M. Sundermeyer, M. Durner, and R. Triebel (2020). *Self-Supervised Object-in-Gripper Segmentation from Robotic Motions*. arXiv: 2002.04487 [cs.CV].

Durner, M., W. Boerdijk, M. Sundermeyer, W. Friedl, Z.-C. Marton, and R. Triebel (2021). "Unknown Object Segmentation from Stereo Images". In: arXiv: 2103.06796 [cs.CV].

Kollar, T., M. Laskey, K. Stone, B. ThananYangjeyan, and M. Tjersland (2022). "SimNet: Enabling Robust Unknown Object Manipulation from Pure Synthetic Data via Stereo". In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. Proceedings of Machine Learning Research. PMLR, pp. 938–948.

Bergmann, P., T. Meinhardt, and L. Leal-Taixé (2019). "Tracking Without Bells and Whistles". In: *The IEEE International Conference on Computer Vision (ICCV)*.

Athar, A., J. Luiten, A. Hermans, D. Ramanan, and B. Leibe (2022). "HODOR: High-level Object Descriptors for Object Re-segmentation in Video Learned from Static Images". In: *CVPR*.

Pont-Tuset, J., F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool (2017). *The 2017 DAVIS Challenge on Video Object Segmentation*.

Kipf, T., G. F. Elsayed, A. Mahendran, A. Stone, S. Sabour, G. Heigold, R. Jonschkowski, A. Dosovitskiy, and K. Greff (2022). "Conditional Object-Centric Learning from Video". In: *International Conference on Learning Representations (ICLR)*.

Cheng, H. K. and A. G. Schwing (2022). "XMem: Long-Term Video Object Segmentation with an Atkinson-Shiffrin Memory Model". In: *ECCV*.

Yang, Z., Y. Wei, and Y. Yang (2021). "Associating Objects with Transformers for Video Object Segmentation". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Voigtlaender, P., M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe (2019). "MOTS: Multi-Object Tracking and Segmentation". In: *CVPR*.

Yang, L., Y. Fan, and N. Xu (2019). "Video instance segmentation". In: *CoRR* abs/1905.04804.

Kim, D., S. Woo, J.-Y. Lee, and I. S. Kweon (2020). "Video Panoptic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Luo, W., J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim (2021). "Multiple object tracking: A literature review". In: *Artificial Intelligence* 293, p. 103448.

Luiten, J., A. Osep, P. Dendorfer, P. Torr, A. Geiger, L. Leal-Taixe, and B. Leibe (2020). "HOTA: A Higher Order Metric for Evaluating Multi-object Tracking". In: *International Journal of Computer Vision*.

Wang, Y., Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia (2021). "End-to-End Video Instance Segmentation with Transformers". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*.

Caelles, A., T. Meinhardt, G. Brasó, and L. Leal-Taixé (2022). "DeVIS: Making Deformable Transformers Work for Video Instance Segmentation". In: *arXiv:2207.11103*.

Wu, J., Y. Jiang, S. Bai, W. Zhang, and X. Bai (2022). "SeqFormer: Sequential Transformer for Video Instance Segmentation". In: *ECCV*.

Carion, N., F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko (2020). "End-to-End Object Detection with Transformers". In: *ECCV*.

Zhu, X., W. Su, L. Lu, B. Li, X. Wang, and J. Dai (2020). "Deformable DETR: Deformable Transformers for End-to-End Object Detection". In: *arXiv preprint arXiv:2010.04159*.

Xie, E., W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo (2021). "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers". In: *CoRR*.

Park, K., S. Woo, S. W. Oh, I. S. Kweon, and J.-Y. Lee (2022). "Per-Clip Video Object Segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1352–1361.

Oh, S. W., J.-Y. Lee, N. Xu, and S. J. Kim (2019). "Video Object Segmentation using Space-Time Memory Networks". In: *ICCV*.

Bewley, A., Z. Ge, L. Ott, F. Ramos, and B. Upcroft (2016). "Simple online and realtime tracking". In: *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 3464–3468.

Wojke, N. and A. Bewley (2018). "Deep Cosine Metric Learning for Person Re-identification". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 748–756.

Meinhardt, T., A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer (2022). "TrackFormer: Multi-Object Tracking with Transformers". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, J., Q. Liu, Y. Jiang, S. Bai, A. Yuille, and X. Bai (2022). "In Defense of Online Models for Video Instance Segmentation". In: *ECCV*.

Kalman, R. (1960). "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering*. D 82, pp. 35–45.

Weber, M. et al. (2021). *STEP: Segmenting and Tracking Every Pixel*.

Xiang, Y., A. Alahi, and S. Savarese (2015). "Learning to Track: Online Multi-object Tracking by Decision Making". In: *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4705–4713.

Teed, Z. and J. Deng (2020). "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow". In: *Computer Vision – ECCV 2020*.

Milan, A., L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler (2016). *MOT16: A Benchmark for Multi-Object Tracking*.

Yan, B., Y. Jiang, P. Sun, D. Wang, Z. Yuan, P. Luo, and H. Lu (2022). "Towards Grand Unification of Object Tracking". In: *ECCV*.

Gordon, D., A. Farhadi, and D. Fox (2017). "Re3: Real-Time Recurrent Regression Networks for Object Tracking". In: *arXiv preprint arXiv:1705.06368*.

Dendorfer, P., H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé (2020). *MOT20: A benchmark for multi object tracking in crowded scenes*.

Zhan, Z., D. McKee, and S. Lazebnik (2022). *Robust Online Video Instance Segmentation with Track Queries*.

Cheng, B., A. G. Schwing, and A. Kirillov (2021). "Per-Pixel Classification is Not All You Need for Semantic Segmentation". In.

Qi, J. et al. (2022). "Occluded Video Instance Segmentation: A Benchmark". In: *Int. J. Comput. Vision* 130.8, pp. 2022–2039. ISSN: 0920-5691.

Wang, W., M. Feiszli, H. Wang, and D. Tran (2021). *Unidentified Video Objects: A Benchmark for Dense, Open-World Segmentation*.

Yang, Z., J. Miao, X. Wang, Y. Wei, and Y. Yang (2022). "Scalable Multi-object Identification for Video Object Segmentation". In: *arXiv preprint arXiv:2203.11442*.

Yang, Z. and Y. Yang (2022). "Decoupling Features in Hierarchical Propagation for Video Object Segmentation". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

He, K., X. Zhang, S. Ren, and J. Sun (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Dosovitskiy, A., L. Beyer, et al. (2021). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ICLR*.

Liu, Z., Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo (2021). "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: *CoRR*.

Wang, W., E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao (2021). "Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction Without Convolutions". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 568–578.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017). "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17, pp. 6000–6010.

Bahdanau, D., K. Cho, and Y. Bengio (2014). *Neural Machine Translation by Jointly Learning to Align and Translate*.

Liang, C., W. Wang, J. Miao, and Y. Yang (2022). "GMMSeg: Gaussian Mixture based Generative Semantic Segmentation Models". In: *Advances in Neural Information Processing Systems*.

Saire, D. and A. R. Rivera (2022). "Global and Local Features Through Gaussian Mixture Models on Image Semantic Segmentation". In: *IEEE Access* 10, pp. 77323–77336.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag. ISBN: 0387310738.

Lin, T.-Y., P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie (2017). "Feature Pyramid Networks for Object Detection". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944.

Ronneberger, O., P. Fischer, and T. Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Springer International Publishing, pp. 234–241.

Denninger, M., M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and H. Katam (2019). "BlenderProc". In: *arXiv preprint arXiv:1911.01911*.

Roboception (2023). *rc_visard 3D Stereo Sensor*. URL: `https://doc.rc-visard.com/latest/pdf/rc_visard_manual_en.pdf`.

Stiefelhagen, R., K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, and P. Soundararajan (2007). "The CLEAR 2006 Evaluation". In: *Multimodal Technologies for Perception of Humans*. Springer Berlin Heidelberg.

Xu, N., L. Yang, Y. Fan, D. Yue, Y. Liang, J. Yang, and T. Huang (2018). *YouTube-VOS: A Large-Scale Video Object Segmentation Benchmark*.

Lin, T.-Y., M. Maire, et al. (2015). *Microsoft COCO: Common Objects in Context*.

Dosovitskiy, A., P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. v. d. Smagt, D. Cremers, and T. Brox (2015). "FlowNet: Learning Optical Flow with Convolutional Networks". In: *2015 IEEE International Conference on Computer Vision (ICCV)*.

Athar, A., J. Luiten, A. Hermans, D. Ramanan, and B. Leibe (2020). *HODOR: High-level Object Descriptors for Object Re-segmentation in Video Learned from Static Images*. URL: `https://github.com/Ali2500/HODOR`.