A Review of Occlusion-Aware Object Representations in Unsupervised Video Understanding

Natasha Sharan

August 23rd, 2025.

Contents

1 Introduction

3D object tracking is a valuable mechanism that lends itself to a variety of applications, including autonomous driving, robotic manipulation, medical imaging, and traffic surveillance. The objective is to detect and localize objects across consecutive video frames while preserving consistent identities over time. Although recent advances in deep learning have produced highly effective supervised tracking models, these approaches rely heavily on large volumes of annotated training data. The collection of these annotations can be both time-consuming and expensive, thus creating a significant bottleneck for real-world applications. Unsupervised and self-supervised approaches aim to overcome this limitation by learning directly from raw video data, eliminating the need for labelled datasets, and allowing for greater scalability. However they struggle with persistent object identification, particularly under challenging visual conditions such as occlusion. When an object is momentarily occluded and reappears, current models often misidentify them, or merge them with another object. This results in degraded tracking performance.

To address these limitations, this work investigates a framework for occlusion-aware unsupervised object tracking that integrates spatial reasoning with temporal dynamics. The key intuition is to represent each video frame as a 3D scene graph, in which nodes correspond to detected objects and edges capture spatial and relational information. Unlike appearance-based features, which can become unreliable during occlusion, the relational structure of a scene often remains informative. For instance, if an object reappears in a spatially consistent position relative to others, its identity can be more reliably preserved.

The proposed framework consists of three main components. The first is an unsupervised depth-aware object detection model that extracts visual features and estimates 3D positions from each frame. This module is implemented in two stages, where initially a depth and pose network are constructed and trained. The object detection module uses cues from the pre-trained depth network to process the frame. The second is a scene graph construction module that organizes this information into an evolving graph structure, where nodes represent objects and edges encode spatial relationships. Lastly, a Graph Neural Network refines the evolving node embeddings over time, capturing spatiotemporal context to enhance tracking performance. The framework is trained in an unsupervised manner, with objectives designed to encourage temporal consistency in embeddings while penalizing unexplained discontinuities in object position or identity.

This report is organized into three additional sections. Section 2 summarizes related works. Section 3 presents a gap analysis, identifying the limitations of existing methods and the architectural differences from the proposed approach. Section 4 will further elaborate on the details of the proposed architecture. Section 5 will analyze results, and Section 6 will conclude this paper.

2 Related Works

This section reviews research related to the development of occlusion-aware object tracking in unsupervised video understanding. It is divided into two parts. Section 2.1 discusses five key papers that are directly relevant to the proposed approach. Section 2.2 surveys additional studies that provide broader context in areas such as self-supervised learning, 3D scene understanding, and the tracking-by-association paradigm implemented through Graph Neural Networks (GNNs). These works collectively highlight the various strategies that have been explored for improving object tracking under challenging conditions including occlusion, and data scarcity. While these methods offer strong performance in their respective settings, they differ from the proposed research in terms of supervision level and architectural design. These distinctions are analyzed in Section 3.

2.1 Key Papers

This section reviews five key papers that are directly relevant to the proposed research on improving occlusion-aware object tracking in videos using 3D scene graphs and GNNs. Each paper contributes foundational ideas that pertain to different aspects of the project, including dynamic spatiotemporal modeling, scene graph generation, multimodal graph reasoning, and occlusion-aware detection. These works demonstrate how spatial and temporal dynamics can be leveraged in low-supervision settings to enhance the robustness of object representations and occlusion-aware tracking. The following subsections present detailed summaries of each paper.

2.1.1 Dynamic GNNs for Occlusion-Aware Tracking

In the paper, "Interpretable Dynamic Graph Neural Networks for Small Occluded Object Detection and Tracking" by Shahriar Soudeep et al. [12], the challenges of occlusion-aware small object tracking are addressed. The research presented in this paper, aims to overcome these limitations by proposing a system that can robustly handle dynamic conditions, thus advancing the capabilities of intelligent transportation systems. The work introduces the DGNN-YOLO framework, which integrates the YOLO11 object detector with Dynamic Graph Neural Networks for robust tracking. The YOLO11 detection mechanism is utilized for its advanced spatial feature extraction capabilities, allowing for the precise initial detection of small objects. Similarly, the DGNN is employed to capture the evolution of object positions, and extract the temporal features of the scene. Fusing the spatial and temporal feature extraction ensures robust object tracking in cluttered scenes.

For each frame, a set of objects are detected by YOLO11, as well as their bounding boxes, the confidence score, the object category, and spatial features. This information is used to construct a dynamic graph, representing the objects and their interactions. The nodes correspond to the detected objects, containing a feature vector consisting of spatial, motion and appearance features. The edges model the relationships between objects and are updated based on the euclidean distance between bounding boxes (proximity), the difference in velocity of objects at a specific time (velocity similarity), and the cosine similarity between feature vectors (appearance similarity). The adjacency matrix encodes the strengths of these relationships. This graph is updated and recalculated dynamically at each frame, ensuring the information is consistently distributed across the nodes. This allows for the mechanism to capture both spatial and temporal dynamics, and ensuring the accuracy of object tracking across frames.

The DGNN-YOLO framework aims to balance the contributions of the detection and tracking tasks, by optimizing two primary loss functions. The detection loss is responsible for evaluating the object detection accuracy, and it is comprised of the sum of a bounding box loss and cross entropy loss. The objective of the tracking loss is to minimize inconsistencies in object identities across frames, which is given by the sum of the squared euclidean distances between nodes, and the cosine similarity of the feature embeddings. The total loss function is the weighted sum of the detection and tracking loss. The YOLO11 component of the model is pretrained, and is further trained and fine tuned on annotated data. Similarly, while the DGNN component learns dynamically, it also relies on ground truth labels, as is evident in the loss functions. Thus it can be noted that the DGNN-YOLO is a supervised model, and requires large amounts of annotated data.

In the evaluation stages, the DGNN-YOLO framework demonstrated significant improvements over existing models such as standard YOLO versions, and Faster R-CNN. It achieved superior results, with a precision of 0.8382, a recall of 0.6875, and a mean average prediction (from 0.5 to 0.95) of 0.6476 on the i2Object Detection Dataset. Without the DGNN component, the performance of YOLO11 was lower, with a precision of 0.8176, exhibiting the necessity of the DGNN and its temporal feature extraction. Similarly, without the spatial features, the performance exhibited a mild decline. These results highlight the importance of both spatial and temporal features for robust object tracking in complex environments. On the other hand, despite its strong performance, the DGNN-YOLO's reliance on visual data causes it to struggle in extreme weather conditions, and it has difficulty differentiating between visually similar or underrepresented classes. The authors have proposed to improve this by integrating sensors such as LiDAR for low-visibility, and exploring semi-supervised learning to improve the detection of rare classes, and reduce dependence on manually labelled data.

2.1.2 Semi-Supervised Temporal GNNs for 3D Tracking

The research paper, "Semi-supervised 3D Object Detection via Temporal Graph Neural Networks" by Jianren Wang et al. [15], proposes a novel framework to address the problem of semi-supervised 3D object detection. The central insight is that spatiotemporal information in 3D point cloud videos can be leveraged to generate more accurate pseudo-labels from unlabelled data through temporal smoothing. These generated labels can then be used to enhance the training of a single-frame 3D object detector. To implement this, the proposed framework employs a teacher-student architecture designed for iterative refinement.

The teacher-network utilizes spatial temporal reasoning to generate high quality pseudo labels on unlabelled data. It consists of two components: an initial 3D object detection module, and a novel spatiotemporal reasoning module. For each frame of the input video, the initial 3D object detector outputs a set of noisy candidate object detections. To refine the results, the novel spatiotemporal reasoning module performs temporal smoothing on the detections, and creates more accurate pseudo-labels. This is accomplished by constructing a video graph using high level extracted features from candidate detections, and then processing it with a graph neural network. The video graph is constructed such that it captures the spatiotemporal relations of the objects in the scene. The nodes are the candidate detections, and the edges link detections across frames based on predicted motion and proximity. The graph is then processed with a graph neural network, which outputs refined predictions. These predictions are then used as pseudo-labels to train the student, a single frame 3D object detector.

This framework incorporates a variety of training methods to optimize each of its components. The teacher network is trained before the student begins its main training iterations, thus establishing an iterative training environment. The initial 3D-detection module is trained on a limited set of labelled data. The GNN is trained end-to-end with Binary Cross Entropy on this same set of data, and its robustness is enhanced through three data augmentation strategies: trajectory-level copy-and-paste, random trim, and random noise. Trajectory-level copy-and-Paste randomly copies trajectories of objects from one video, to another video. Random trim randomly selects the first and last frame from a video to form a new sequence of frames, and random noise injects noise to the labelled information. These augmentations help prevent the GNN from overfitting to the small labelled dataset. The student shares the same architecture as the teacher's 3D detection module, and is initialized with the same parameters. It is trained on a combination of the original limited labelled data, and the pseudo-labelled data generated by the teacher. Since the pseudo-labels may not be accurate, an uncertainty-aware training mechanism is used to mitigate their impact. The GNN's output scores for pseudo labels are first calibrated using histogram binning. The entropy of these calibrated scores is calculated to estimate the uncertainty of each pseudo-label. This uncertainty value is then used to weight the classification and regression losses for the student network, reducing the impact of less reliable pseudo-labels.

The continual improvement of this framework is achieved by combining gradual semi-supervised training with the iterative refinement of the student and teacher. During the initial stages of training the student, the generated pseudo-labels will contain noise and can be unreliable. However as the teacher improves, the pseudo-labels will become more accurate. To accommodate this behaviour, the amount of unlabelled data used for training will gradually increase across iterations, allowing for the student to learn from larger amounts of increasingly reliable pseudo labels. After the student is trained in an iteration, the parameters of the teacher's object detection module are updated using an exponential moving average of the student's updated parameters. This allows the teacher to benefit from the student's improvements. The GNN is then re-trained on labelled data for a set number of epochs, allowing for it to adapt to the improved candidates generated by the updated 3D object detector. This iterative refinement and gradual approach allows for both student and teacher to improve their performance, thus continually improving the framework.

The proposed framework demonstrated a superior 3D object detection performance compared to baselines trained with the same limited labelled data. On the nuScenes dataset, it achieved a mean average precision (mAP)of 0.3646, outperforming both the baseline (0.2317) and another semi-supervised model SESS (0.2745). Similarly on the H3D dataset, it achieved 0.3899 mean average precision compared to the baseline's 0.3102. Ablation studies confirm that each component of the framework meaningfully contributes to the high performance. However, while this framework demonstrates its proficiency in 3D object detection, the paper acknowledges the performance gap between the semi-supervised model (0.3899 mAP) and a fully supervised model (0.4556 mAP). Additionally, classes with more labelled data appeared to benefit less from semi-supervised learning.

2.1.3 GNNs with Cross-Edge Modality Attention

The paper, "3D Multi-Object Tracking Using Graph Neural Networks with Cross-Edge Modality Attention" by Martin Buchner and Abhinav Valada [1], introduces Batch3DMOT, a novel multimodal GNN framework for offline 3D multi-object tracking. This work aims to provide highly accurate offline tracking, suitable for generating pseudo ground-truth labels. A core limitation addressed in this research is modality intermittence, where sensor data like LiDAR can be sparse or missing for objects, thus complicating feature representation and reducing tracking robustness. Batch3DMOT approaches these challenges by employing a novel agglomerative trajectory clustering scheme for effective pseudo trajectory generation, as well as a cross-edge attention mechanism that utilizes fragmentary sensor data.

The proposed framework follows the tracking-by-detection paradigm, where objects are initially detected, and a tracking module associates the detections over time to form consistent object trajectories. For each frame, Batch3DMOT collects 3D object detections from multiple sensor modalities such as camera, LiDAR, and radar. To model associations across time, these detections are used to construct a directed, acyclic, and category-disjoint graph over a sliding window of 5 frames. The nodes in this graph represent detections, and are associated with a fusion of a primary "3D Pose and Motion" (3D-PM) feature, and three additional modality-specific features: 2D appearance (2D-A), 3D LiDAR appearance (3D-AL), and Radar Features (3D-R). The primary node feature consists of spatiotemporal information, and the modality-specific features consist of encoded visual information from sensors. Edges in the graph are directed in a forward-time manner, where nodes across different timestamps are connected, only if they belong to the same object category. This construction results in the formation of separate graph components for each category. Connections between components are established using a normalized kinematic similarity metric, encouraging robust associations while distinguishing between categories.

To learn the temporal associations, the framework employs a multimodal GNN that performs time-aware neural messages passing across nodes and edges, with several novel attention mechanisms. The cross-edge modality attention module dynamically weighs modality-specific features for every edge. This allows the model to adaptively assess the reliability of modalities when comparing detections, thus mitigating the problem of modality intermittence. Additionally, the inter-category graph attention enables information exchange between disconnected category subgraphs by constructing temporary framewise graphs. The framewise graphs are constructed by connecting each node to its top-k neighbours in the frame, regardless of category, and are then updated with a graph attention layer. These attention mechanisms allow the GNN to reason over missing or ambiguous detections while maintaining category aware structure. The final output trajectories are generated using a novel agglomerative trajectory clustering scheme, which clusters detections into trajectories according to temporal consistency and edge strength.

The GNN is trained to optimize the detected object's temporal associations, specifically the training objective is to classify whether or not graph edges belong to valid object trajectories. This requires edge labelling, where an edge is classified as active (valid) if it represents the closest time-wise occurrences of the same object, otherwise it is inactive (invalid). These are generated by matching actual object detections to ground truth trajectory annotations. Due to the category-disjoint graph structure and the varying frequencies of object categories, the GNN is trained with a class-balanced binary cross-entropy loss. The loss function weighs edges according to the frequency of their respective object categories, thus minimizing the influence of category imbalance. The framework is trained over 100 epochs, and uses six rounds of message passing. The modality-specific encoders, used to encode the nodes' modality-specific features, are trained separately. This modular training approach allows the GNN to focus on spatiotemporal association learning, while the modality encoders learn to extract discriminative object features.

Batch3DMOT demonstrates superior results when evaluated on the nuScenes and KITTI benchmarks, compared to prior methods in Average Multiple Object Tracking Accuracy(AMOTA). The 3D-PM-CL variant, which combines pose and motion features with camera and LiDAR data, scored 0.689 AMOTA on the nuScenes test set, outperforming OGR3MOT by 3.3%. Ablation studies confirm the necessity of the Batch3DMOT's novel components. Specifically, the cross-edge modality attention demonstrated significant improvements and the agglomerative clustering strategy outperformed simpler heuristics. Qualitative results demonstrate superior false positive suppression and improved tracking of stationary objects. Despite its performance, the paper addresses the limitation of long-term occlusion handling due to its 5 frame processing window.

2.1.4 Iterative Scene Graph Generation

The paper "Iterative Scene Graph Generation" by Siddhesh Khandelwal and Leonid Sigal [6], introduces a novel transformer-based framework for generating scene graphs through an iterative refinement process. The framework aims to enable more consistent and context-aware relational understanding, by collectively updating predictions of subjects, objects and predicates overtime. The iterative refinement strategy is modelled by initially producing a scene graph estimate using a traditional factorization, where the graph can be represented as a set of triplets, (subject, object, predicate). Then over multiple refinement steps, this graph is systematically improved, where each refinement is conditioned on the previously generated graph. By conditioning across refinement steps, it compensates for conditional independence assumptions, thus allowing for the framework to jointly reason over the subjects, objects and predicates.

The proposed framework is implemented using a novel transformer architecture, consisting of an image encoder, and predictor decoders. The image encoder is built upon the DETR object detection framework, employing a deep convolutional network and a multi-layer transformer encoder. The predictor decoders each predict the individual components of the relationship triplets, subject, object, predicate, and are implemented as three separate, synchronized, multi-layered transformer decoders. Each layer in these decoders corresponds to one refinement step, and using the image features and queries produced by the encoder, they generate a scene graph estimate.

For each frame, the image encoder initially obtains an image-level spatial map using its CNN component, and the encoder transformer component converts it into a position-aware flattened image feature representation. The framework then enters an iterative loop, where each loop corresponds to a layer in the decoders, thus also corresponding to a refinement step. The predictor decoders take a set of input queries, which are then modified by two conditioning methods to enable joint reasoning. The first allows for conditioning decoders within the same step, where the object decoder is conditioned on the subject decoder's output, and the predicate decoder is conditioned on both the subject and object decoders' output. The second method involves all three decoder layers to be conditioned on the outputs generated by the previous step. The decoders use attention mechanisms to process these conditioned inputs along with the image features, thus producing a more refined set of triplet predictions which form the scene graph estimate for that step. The output from the final refinement step undergoes a non-maximum suppression strategy, grouping the predictions into unique entity instances, thus producing the final scene graph.

Training the framework end-to-end is performed using a novel joint loss function. It is applied at each refinement step, ensuring that the model produces a valid and coherent scene graph throughout the iterative process. At each step, the model outputs a fixed-size set of predicted relationship triplets, then using the Hungarian algorithm for bipartite matching, these predictions are matched to the variable-size ground truth set. A single optimal matching permutation is computed jointly across all refinement layers, thus introducing implicit dependencies between the subject, object and predicate decoders. The matching cost for each candidate triplet, includes both a classification term, based on the negative dot product of predicted and true class distributions, and a bounding box regression term, which combines L1 loss and Generalized IoU loss for the subject, object, and predicate. After finding the optimal permutation, the total loss is calculated as the sum of the losses from all refinement steps, with each step using this same matching to supervise classification and localization. To address the class imbalance present in scene graph datasets, the training process incorporates a loss re-weighting scheme.

The framework was evaluated on two major datasets, and demonstrated strong accuracy, adaptability, and generalizability. On the Visual Genome dataset, the model outperformed existing one-stage and transformer-based baselines like SGTR in terms of Recall (R@K), Mean Recall (mr@K), and Harmonic Recall (hR@K). Similarly, when evaluated on the Action Genome dataset, the framework achieved a 12.6-point gain in mR@20 and a 3.1-point gain in R@20 over the closest baseline These results highlight the effectiveness of the model's iterative refinement and loss re-weighting strategies in improving generalization across predicate distributions. Additionally, to demonstrate the generality of the iterative refinement formulation, the framework was applied as an augmentation to an existing two-stage model (MOTIF). This resulted in the consistent improvement of its performance with each refinement step, increasing its mR@20 and mR@50 scores by approximately 6% after three steps. Despite the strengths of the framework, it was noted that the use of a shared image encoder across refinement layers could limit representational diversity, and due to the DETR-style detection, the task of small object detection remains a challenge.

2.1.5 Context-Aware Compositional Nets

The paper "Robust Object Detection under Occlusion with Context-Aware CompositionalNets" by Angtian Wang, Yihong Sun et al. [13], introduces a novel detection framework that significantly improves object detection performance in heavily occluded scenes. The proposed framework, Context-Aware CompositionalNets (CA-CompositionalNets), extends the architecture of the original CompositionalNets by integrating two core innovations. The first is context-aware representation which enables the model to explicitly disentangle the representation of a detected object from its surrounding context, thus preventing it from being misled by contextual biases in training data. The second is the robust part-based voting mechanism, which allows for the model to leverage visible parts of an object to vote for key points of its bounding box. These extensions demonstrate effectiveness in detecting objects robustly under severe occlusion.

The framework applies a CNN backbone to extract a feature map from the input image frame. This map is then processed by a detection layer that generates a spatial response map representing the likelihood of the object center at each location. Non-maximum suppression is then used to select initial candidate object centers. Context-awareness is integrated during training by segmenting feature activations into object and context regions based on the ground truth bounding boxes. Features with receptive fields falling outside the bounding box are treated as context and clustered to build a dictionary of common context patterns. A generative mixture model is then trained to separate object and context contributions using von-Mises-Fisher (vMF) distributions, with a weighting parameter controlling the relative influence of context and object likelihoods. At inference time, this model is used to compute a combined likelihood score for each feature, where high object-likelihood indicates that the feature is a reliable part of the object. This mechanism allows the model to suppress false positives caused by misleading contextual cues thus improving robustness to occlusion.

Using the set of reliable object-part features identified by the mixture model, the framework employs a part-based voting mechanism to estimate the bounding box of the object. For each reliable feature at a spatial location, the model uses learned spatial offset templates derived from the compositional model component to cast probabilistic votes for three key locations: object center, bottom-left corner, and top-right corner. These offset templates capture the expected relative positions between each part and the target locations, and are conditioned on object pose and category. The votes are accumulated into three separate spatial heatmaps corresponding to the object-center, bottom-left, and top-right corners. The magnitude of each vote is determined by the feature's object-likelihood as computed by the mixture model, thus ensuring only high-confidence parts contribute meaningfully to localization. After aggregation, the peaks of the corner heatmaps are used to determine the bounding box, and the center heatmap is used to align detections. This mechanism allows for robust detection under occlusion by leveraging the spatial consistency of visual parts.

Training is performed end-to-end using a joint objective that combines a classification loss, a generative modeling loss, and a keypoint-based detection loss. The classification loss is a cross-entropy loss with an L_2 weight regularization term, thus encouraging the model to accurately classify object categories while preventing overfitting. The generative modeling loss supervises the learning of compositional part features using von-Mises-Fisher (vMF) distributions and consists of two components. One component learns directional clusters over object part features, while the other incorporates a context mask to supervise the separation of object and context features. The detection loss ensures the model can accurately localize the object and its bounding box keypoints: object center, bottom-left corner, and top-right corner. For each of the three key locations, the model generates a spatial response map. The Dice loss is applied to each of these maps to maximize the similarity with the corresponding ground truth heatmaps. The total loss is a weighted sum of the three loss components, where the weights balance classification accuracy, context-aware representation learning, and localization performance.

The framework was evaluated on two datasets: the synthetic OccludedVehicles dataset based on PASCAL3D+ and the real-world OccludedCOCO dataset. For the most heavily occluded objects, the model outperforms Faster R-CNN by 41% in mean average precision (mAP) on the OccludedVehicles dataset and by 35% on the OccludedCOCO dataset. Ablation studies validate the importance of both the context-aware mixture model and the part-based voting mechanism. Removing the context-disentanglement component results in more frequent misclassification of occluded objects, thus highlighting the significance of the mixture model. Similarly, replacing the voting mechanism with a Region Proposal Network (RPN) results in a substantially worse localization performance, demonstrating the importance of the robust bounding box voting mechanism. Overall, the results demonstrate a significant improvement over baseline methods, particularly under moderate to heavy occlusion. However, in cases of light or no occlusion, the model's high sensitivity to bounding box alignment causes a slightly lower performance than Faster R-CNN. Additionally, the architectural complexity and generative modelling results in a higher computational cost compared to standard detectors.

2.2 Additional Studies

Recent advances in unsupervised and self-supervised learning have contributed significantly to the development of object tracking systems that operate without annotated data. This research is particularly relevant for addressing challenges such as occlusion, long-term temporal association, and complex scene dynamics. The following subsections explore areas that directly support the objectives of the proposed research, which focuses on unsupervised occlusion-aware object tracking using 3D scene graphs and GNNs.

2.2.1 Unsupervised and Self-Supervised Tracking

Unsupervised and self-supervised tracking methods aim to learn object motion and appearance representations without requiring manually annotated labels. These approaches are allow for reduced data annotation costs and scalable tracking in unconstrained environments. This subsection reviews three frameworks that leverage temporal consistency, contrastive representation learning, and pseudo-label generation to improve tracking accuracy under fully unsupervised settings. Each method is evaluated on standard tracking benchmarks and demonstrates competitive or superior performance compared to supervised baselines, despite relying entirely on unlabeled video data.

Shyamgopal Karthik et al. [5] introduce SimpleReID, a fully unsupervised framework for learning object re-identification models from unlabeled videos. The framework employs the unsupervised SORT tracker to initially generate noisy tracklets without any appearance features. These tracklets are then used as pseudo-labels to train a ResNet-50 model, with the objective of classifying image patches into tracklet labels. The model is trained using a standard cross-entropy loss, under the assumption that tracklets across videos are independent, and tracklets within the same video mostly correspond to distinct individual objects. For evaluation, SimpleReID replaces supervised ReID modules in existing tracking frameworks without modifying other components. When combined with CenterTrack and evaluated on the MOT16 and MOT17 benchmarks, the model outperformed all trackers on public detections. It outperformed the previous best tracker, the original CenterTrack, by 0.2 and 0.3 MOTA points, and 4.4 and 4.8 IDF1 points, respectively on MOT16 and MOT17. Limitations include a heavy reliance on noisy labels, and assumptions about tracklet uniqueness which may be violated due to occlusions.

Ning Wang et al. [16] present the Unsupervised Deep Tracker (UDT), an unsupervised visual tracking framework built on a Siamese correlation filter network that leverages forward-backward consistency as a self-supervisory signal. The methodology consists of two stages: forward tracking and backward verification. Starting from a randomly initialized target region, the model tracks it forward through video frames and generates a response map that indicates the target's new location. The location with the maximum response in the map is used as a pseudo label, which then serves as a starting point for tracking the target region back to the initial frame. A consistency loss is computed using the difference between the backward-predicted location and the initial starting position. This loss is further refined by a multiple-frame validation strategy to amplify tracking errors, which encourages effective penalization of inaccuracies, as well as a cost-sensitive weighting scheme that downweights noisy samples and emphasizes informative ones.The framework is trained end-to-end using SGD on the ILSVRC 2015 dataset, requiring no manual labels or pre-processing. Evaluation on benchmarks like OTB-2015, Temple-Color, and VOT-2016 shows that UDT matches the performance of baseline supervised trackers. However, the model struggles with severe appearance changes due to the lack of supervised object-level features and incurs a higher computational cost during training because of its forward-backward tracking loop.

Qiangqiang Wu et al. [17] propose the Progressive Unsupervised Learning (PUL) framework that learns a robust feature representation for tracking in two main stages: Background Discrimination (BD) learning and Temporal Correspondence (TC) learning. In the first stage, the model uses contrastive learning to distinguish objects from the background using static frames. This method is enhanced using a proposed Anchor-based Hard Negative Mining (AHM) strategy where challenging negative samples are heuristically selected to improve the model's discriminative ability. In the second stage, TC learning utilizes the BD model to process short, unlabeled videos to mine corresponding patches over time. The patch from the first frame and the tracked patch from the last frame are collected as a training pair. Since the mined pairs are noisy, a novel Noise-Robust (NR) loss function is employed, such that spatial noise is explicitly modeled in the tracked patches, enabling the network to learn temporal consistency from noisy data. The framework is trained completely offline and end-to-end using unlabeled data from the ILSVRC-2015 dataset, first training the BD model, then mining patch-pairs for TC learning with a combined loss function. Evaluation on multiple benchmarks shows that PUL significantly outperforms other unsupervised trackers and competes closely with supervised baselines. However, the model has a limitation with long sequences, as it experiences more tracking failures.

2.2.2 Graph-Based Tracking Using Learned Associations

Graph-based approaches are useful for object tracking tasks, as they are able to model complex interactions between objects in a scene. This subsection reviews a representative framework that demonstrates how learning graph-structured associations can significantly improve tracking performance in online settings.

Ioannis Papakis et al. [11] introduce an online tracking-by-detection framework that formulates data association as a bipartite graph matching problem between existing tracklets and new detections. It extracts appearance and geometric features for each object and constructs a graph where a Graph Convolutional Neural Network (GCNN) learns interaction features to model spatial context. Affinity scores between tracklets and detections are computed from these features and normalized using the Sinkhorn algorithm, which directly enforces one-to-one matching constraints in a differentiable, end-to-end fashion. The model is trained using a weighted binary cross-entropy loss over the Sinkhorn-normalized association matrix, and a randomized sampling strategy that draws tracklet frames from up to 45 frames in the past to simulate difficult occlusion scenarios. Evaluated on the MOT15–MOT20 benchmarks, GCNNMatch achieved state-of-the-art accuracy among online trackers, significantly outperforming strong baselines like Tracktor-v2 and other GNN-based models in both MOTA and IDF1 metrics. However, the framework relies on the last observed instance of each tracklet and pre-processed detections, which limits its robustness under long-term occlusion and constrains its performance to the quality of the detector.

2.2.3 Occlusion Handling Mechanisms

Occlusion poses a significant challenge in object tracking and motion estimation, often leading to identity switches, tracking failures, and degraded performance. Recent works address this problem by employing a variety of strategies, including geometric modeling, occlusion-aware graph reasoning, classifier memory, and motion-appearance disentanglement. This subsection reviews four frameworks that introduce occlusion-aware modules or mechanisms to enhance robustness in object tracking and motion estimation. Empirical results on standard benchmarks, along with detailed ablation studies, demonstrate the significance of occlusion handling mechanisms in object tracking tasks.

Guangming Wang et al. [14] present a framework for joint unsupervised learning of depth, pose, and optical flow from unlabeled monocular video sequences. The main contribution is an explicit occlusion handling strategy based on 3D geometric projections. The framework segments each frame into three distinct regions: rigid, non-rigid, and occluded. Motion in the rigid regions results from camera movement, while the non-rigid regions account for independently moving objects. The occluded regions are filtered out using a novel parameter-free occlusion mask, composed of edge, overlap and blank masks. To further improve robustness, a novel less-than-mean (LTM) mask is used to filter outlier pixels influenced by motion or illumination changes, while a new maximum normalization strategy ensures stable depth estimation in textureless areas. For optical flow, a novel consistency loss is proposed, where the framework leverages depth and pose to generate a rigid flow which can provide supervision in occluded regions. The depth and pose networks are trained jointly, using reconstruction and smoothness losses. These are then used to help train the optical flow network, which uses reconstruction, smoothness, and consistency losses across multi-scale inputs. After training on the KITTI and Cityscapes with no ground truth labels, the method outperforms prior unsupervised approaches on all three tasks and is competitive with stereo-supervised methods. However, a limitation can result from the reliance on rigid flow for supervising occluded regions, particularly in scenes with non-rigid motion.

Qiankun Liu et al. [9] introduce two novel modules that can be integrated into existing online, tracking-by-detection Mutli-Object-Tracking (MOT) systems. The first is an unsupervised Re-ID learning module, which learns to produce discriminative appearance features for objects without any identity labels. It leverages the similarity of object appearances across adjacent frames, and dissimilarity within the same frame, which is implemented using a matching-based loss objective. Specifically, a similarity matrix is computed between all objects in two adjacent frames, and the model is optimized using a combination of inter-frame, intra-frame, and cycle consistency losses. The second module is an Occlusion Estimation Module which detects likely occlusion regions as keypoints using a CNN trained on automatically generated occlusion maps based on overlapping bounding boxes. This enables the system to recover lost tracklets by reasoning about occluders and predicted object motion. The module is trained with a supervised occlusion loss, consisting of heatmap and offset losses derived from bounding box overlap. The modules are integrated into two state-of-the-art MOT frameworks, FairMOT and CenterTrack, and they are trained jointly. The frameworks are evaluated on MOT16, MOT17, and MOT20, and demonstrate strong performance. The unsupervised Re-ID module matches or exceeds supervised baselines in IDF1 and MOTA, while the occlusion module significantly reduces false negatives. However, the occlusion module still relies on ground truth bounding boxes for supervision, and both occlusion handling and refinding remain sensitive to hyperparameters and motion model assumptions.

Xingping Dong et al. [3] propose a real-time, occlusion-aware tracking-by-detection algorithm that employs different strategies depending on the occlusion level of each frame. The framework consists of two key components: a two-stage classifier based on Integrated Circulant Structure Kernels (ICSK), and a classifier-pool module for occlusion handling. In the first stage, the ICSK performs position estimation using kernelized ridge regression over circulant matrices, and is accelerated by Fast Fourier Transform (FFT). The second stage uses a scale estimation strategy based on Discriminative Scale Space Correlation Filters (DSSCF). The classifier-pool module is implemented as a FIFO queue that stores recent reliable classifiers trained on non-occluded frames. When occlusion is detected, the tracker selects the most appropriate classifier from the pool using an entropy minimization criterion. It also extracts Histogram of Oriented Gradients (HOG) and Colour Name (CN) features, and applies different update strategies depending on the occlusion status. Training is conducted online, where the ICSK is updated frame-by-frame using ridge regression, and DSSCF is used for scale learning. The classifier-pool module selects classifiers using an energy function that combines log-likelihood of the highest response and entropy regularization to favour confident classifiers. If occlusion is detected, classifier updates are skipped to preserve the reliability of previously learned models. The framework was evaluated on 56 challenging videos, and outperformed other leading trackers in accuracy and robustness, while operating at a real-time speed of 29 fps. However, the method depends on empirically set parameters and uses handcrafted features (HOG, CN), which may limit generalizability and performance compared to deep learning-based trackers.

Yubo Zhang et al. [20] introduce a graph-based framework for tracklet-detection association that dynamically adjusts feature aggregation based on object occlusion. The proposed framework consists of an Occlusion-Related Graph Convolutional Neural Network (OR-GCNN), upon which an Interactive Similarity Module (ISM) is built. Using bounding box positions obtained by a backbone detector and appearance features extracted with a Re-ID network, the ISM computes interactive similarity scores used for tracklet-detection association. For each node representing a detection, its occlusion weight is derived from the maximum Intersection over Union (IoU) it shares with other objects in the same frame. This weight modulates the OR-GCNN's update rule, and influences how much the model relies on an object's appearance features compared to that of its neighbours. In cases of high occlusion, the model relies more heavily on neighbouring features, otherwise self-features are prioritized. The final interactive similarity between each tracklet-detection pair is computed using cosine similarity between their updated node features. The ISM is trained offline using weighted binary cross-entropy loss, where the objective is to learn a matching function that can distinguish between correct and incorrect tracklet-detection pairs. The trained module is then integrated into an existing tracker, BoT-SORT, as a fixed component. It demonstrates a strong performance when evaluated on MOT16 and MOT17, achieving MOTA scores over 80 and outperformed baselines such as ByteTrack and BoT-SORT. However the framework increases association time, is sensitive to detection errors, and relies on manually tuned parameters.

2.2.4 Dynamic 3D Scene Understanding

The task of dynamic 3D scene understanding is highly relevant to a variety of applications including object detection. In an unsupervised setting, this involves learning from raw or partially structured 3D data without explicit semantic labels. Recent works approach this task by introducing frameworks for reconstructing object geometry, detecting functional elements, segmenting scenes, and separating dynamic and static components. This subsection reviews six methods that advance unsupervised or minimally supervised 3D scene understanding, through geometric priors, self-supervised learning, probabilistic occupancy modeling, and multimodal reasoning.

Geonho Cha et al. [2], introduce 3D-URN, an unsupervised deep learning framework that reconstructs 3D object structures from 2D feature points under an orthographic camera model. The framework consists of two main components: a 3D shape reconstructor module, and a rotation estimator module. The 3D shape reconstructor is composed of multiple neural 3D reconstructor modules, each of which take vectorized 2D inputs and produce a 3D output using multiple cascaded fully connected layers. The final 3D shape is generated as a weighted sum of the outputs from these reconstructors, with the weights being determined by a separate weight estimator network. The rotation estimator component uses the 2D feature points to estimate a 3D rotation matrix, which is further refined by performing differentiable Singular Value Decomposition (SVD) to enforce orthogonality and ensure validity. The framework is trained using a combination of three unsupervised loss components: projection loss, low-rank prior loss, and regularization. The projection loss measures how well the 3D reconstruction projects back to the original 2D points using the estimated rotation. The low-rank prior loss encourages a low-dimensional shape space across the batch by minimizing the nuclear norm of 3D reconstructions. A regularization term penalizes large weights from the estimator to avoid overfitting. The total loss is a weighted sum of these components, and supervision is applied at intermediate layers to improve training stability. The framework was evaluated on the PASCAL3D+ and MUCT datasets, demonstrating strong generalization, and handling both rigid and non-rigid objects. However, the framework does not handle missing or occluded keypoints, and lacks temporal modeling for sequential-based inputs.

Kevin Lai et al. [7] propose Hierarchical Matching Pursuit for 3D Data (HMP3D), a hierarchical unsupervised feature learning framework for semantic scene understanding. It assigns object class labels to every 3D point in an RGB-D reconstructed scene by integrating features from both 2D image frames and 3D voxelized point clouds. The framework is built around a Markov Random Field (MRF) that combines predictions from two classification branches and enforces smooth labeling over neighbouring voxels. The first branch is the HMP3D module, which consists of voxel classifiers that learn hierarchical sparse features directly from 3D data using a two-layer Hierarchical Matching Pursuit (HMP) network. It constructs a two-layer hierarchical sparse feature representation by learning dictionaries over 3D voxel patches at different scales and applying 3D spatial pyramid max pooling, thus enabling robustness to local shape variations and scale. The second branch consists of RGB-D image classifiers, which learn sparse dictionaries over grayscale and depth image patches, and apply sliding-window detection across video frames. The outputs from both branches are projected into 3D and incorporated into the MRF's data term, while its pairwise term penalizes label inconsistencies based on local geometric cues such as surface normal similarity and concavity. The final object labels are inferred over the voxel grid using graph cut optimization. The framework is trained unsupervised, where the HMP3D component is trained on synthetic 3D scenes built from CAD models rendered from multiple viewpoints, while the Image Classifiers are trained on real RGB-D object data. The framework demonstrates a strong performance when evaluated on the RGB-D Scenes Dataset v2 and the Princeton Shape Benchmark, outperforming hand-crafted 3D descriptors. However the method relies heavily on pre-processing such as accurate 3D reconstruction and region-growing segmentation, and struggles when depth data is missing or incomplete.

Vitor Guizilini et al. [4] present PS-HM, a two-step unsupervised feature learning framework for identifying planar surfaces from an unorganized 3D point cloud. These learned features are then integrated into the Hilbert Maps framework for 3D scene reconstruction. The first step is spatial clustering, which employs a novel Automatic Stop K-means (ASK-means) heuristic that automatically determines the number of clusters while outperforming K-means++ in speed. The second step is orientation clustering, where for each initial cluster, a normal vector is calculated from the eigenvector corresponding to the smallest eigenvalue of its covariance matrix, thus capturing the cluster's orientation. The clusters are then iteratively merged according to angular similarity between normals and spatial proximity, to form larger clusters that represent large planar regions. These learned feature clusters are then modelled using a novel Planar Surface covariance function which is integrated alongside standard kernels within the Hilbert Maps framework to construct a probabilistic 3D occupancy model. The entire framework is trained unsupervised, where the final set of learned features are passed into the Hilbert Maps, whose weights are then learned using online stochastic gradient descent, using occupied and unoccupied samples. Upon evaluation, PS-HM significantly reduces memory usage, and achieves a higher classification accuracy than baseline methods such as Octo-Map and LARD-HM, especially in sparse or structured indoor environments. However, the method is specifically designed for planar surfaces, thus limiting its effectiveness in unstructured scenes.

Feixiang Lu et al. [10] introduce a novel framework for real-time reconstruction of dynamic objects and static environments using a single moving depth camera. The approach explicitly distinguishes dynamic objects from the background and reconstructs them simultaneously and separately across three main stages. The first stage continuously segments the scene into static and dynamic regions by classifying spatial K-means clusters of 3D depth points using an energy minimization function, thus producing a trimap that labels regions as dynamic, static or uncertain. The energy function consists of a data term that matches raycasted depth from prior frames, a regularization term for spatial smoothness across clusters, and a context term which enforces temporal consistency using raycasted context maps. In the second stage, a sequential 6D pose prediction module is employed to robustly handle fast motion by using a constant acceleration model, which predicts the next-frame pose using delta transformation vectors computed over previous frames. This predicted pose serves as the initial estimate for a point-to-plane Iterative Closest Point (ICP) algorithm, thus improving registration robustness and convergence speed. In the third stage, the trimap is used to maintain two separate Truncated Signed Distance Function (TSDF) volumetric models: a low-resolution model for the static environment, and a high-resolution DynamicFusion-based model for the non-rigidly moving object. These volumes are then raycasted to generate a synthetic depth map that guides tracking and a probabilistic context map to enforce temporally consistent segmentation. The framework does not rely on pre-trained models or scene priors, and instead adapts online to a stream of depth maps, using optimization-based fusion and motion tracking. Evaluated on multiple real-world scenes, the framework outperforms KinectFusion and DynamicFusion by accurately reconstructing both dynamic and static components at a real-time speed of about 16 fps. However, the approach assumes consistent object topology and smooth motion, and it is sensitive to poor-quality or noisy depth data.

Chenyangguang Zhang et al. [19] present OpenFunGraph, an open-vocabulary framework designed to capture functional relationships in a scene by leveraging the knowledge embedded in Visual Language Models (VLMs) and Large Language Models (LLMs). In particular, the aim is to construct functional 3D scene graphs, a novel extension of traditional 3D scene graphs, which enable functional reasoning by jointly modelling objects, interactive elements and their functional relationships. The implementation extends over four main stages: node detection, node description, functional reasoning, and graph construction. In the detection stage, object and interactive element candidates are initially detected from input frames using RAM++ and GroundingDINO. These 2D candidates are back-projected into 3D using camera poses and depth masks, and are then fused across multiple views to construct a final set of 3D node candidates. In the second stage, multi-view captions from LLAVA are aggregated and summarized with GPT-4 to generate detailed descriptions of each 3D node. The third stage uses a sequential, chain-of-thought reasoning approach to infer functional relationships between nodes, by first identifying local relationships using spatial filter and LLM-based logic, and then inferring remote relationships by having an LLM propose likely connections whose feasibility is assessed by a VLM. A final confidence score is then assigned to each potential remote link. In the fourth stage, nodes and functional edges are connected to construct a complete scene graph. The framework requires no task-specific training, instead relying on prompted inference from pre-trained foundational models. Evaluated on a novel dataset presented in this paper, FunGraph3D, and SceneFun3D, the framework significantly outperforms adapted baselines, including ConceptGraph and Open3DSG, especially in its ability to recognize interactive elements and correctly infer functional connections. However, the method has limitations due to its reliance on external model accuracy, and lack of precision-based evaluation metrics.

Jiaxu Liu et al. [8] propose U3DS^3, a novel framework for fully unsupervised semantic segmentation of holistic 3D scenes that leverages the intrinsic structure and geometry of point clouds. The framework is built on a two-pathway network that learns robust feature representations using the principles of colour invariance and geometric equivariance, and employs an iterative clustering-based refinement training strategy. The pre-processing step creates superpoints to guide pseudo-label assignment, by using Voxel Cloud Connectivity Segmentation (VCCS) to over-segment the input point cloud and merging the resulting segments based on cosine similarity to form larger semantic-aligned regions. The voxelized input point cloud is then processed through two parallel pathways that share a 3D CNN architecture. Each pathway applies different augmentations to enforce feature consistency, both apply distinct colour transformations, and one additionally applies a geometric transformation followed by its inverse. Features are then devoxelized using trilinear interpolation to preserve fine-grained point-level detail. The framework is trained with an unsupervised iterative approach inspired by DeepCluster, where the features produced by the two pathways undergo mini-batch k-means clustering, guided by superpoints, to assign pseudo-labels. These labels supervise the network in subsequent iterations, thus gradually refining feature representation. The loss function promotes pathway consistency by minimizing cosine distance between point features and cluster centroids, and includes two terms. The first is an intra-pathway clustering loss which matches features to pseudo-labels from the same pathway, and a cross-pathway clustering loss, which aligns features to labels generated by the alternate pathway. U3DS^3 achieves strong performance on S3DIS, ScanNet, and SemanticKITTI, outperforming prior approaches like GrowSP on key benchmarks. However, the voxel-based representation may limit spatial precision, and there remains a considerable performance gap relative to supervised methods.

2.2.5 Self-Supervised Representation Learning for Graph Neural Networks

Advances in self-supervised representation learning for GNNs can offer a potential foundation for object tracking in settings without annotations. Since spatiotemporal relationships between objects can be naturally modeled as graphs, self-supervised GNNs enable the learning of discriminative and temporally consistent representations from raw visual data. This subsection reviews a framework that advances self-supervised GNN training through latent graph prediction and highlights its relevance to downstream tracking applications.

Yaochen Xie et al. [18] introduce LaGraph, a predictive self-supervised learning framework that trains GNNs using a theoretically grounded latent graph prediction task. For a given input graph, a masked variant is constructed by randomly masking features of a subset of nodes. Both graphs are encoded by a shared GNN encoder to produce node representations. A decoder reconstructs the original node features from the unmasked graph's representations. Since the latent node features are not observed, LaGraph optimizes a derived self-supervised upper bound on the true latent prediction loss, combining a reconstruction term and an invariant regularization term. The reconstruction term ensures the learned representations are informative, while the invariant regularization term encourages the model to rely more on contextual information from the node's neighbourhood, rather than just its own features. The framework is trained end-to-end, where the encoder and decoder are jointly optimized using a weighted sum of the reconstruction and invariance losses on unlabeled graphs. After pretraining, the encoder is frozen and evaluated by training a linear classifier on its output representations for downstream tasks. LaGraph achieves strong performance across several node and graph-level benchmarks, demonstrating robustness to small batch sizes and limited memory. However, it currently models only node features and not latent edge structures.

3 Gap Analysis

This section analyzes the methodological gaps between the reviewed works and the proposed framework for unsupervised, occlusion-aware 3D object tracking using scene graphs and Graph Neural Networks (GNNs) introduced in Section 4. This comparison highlights central gaps across existing methods, including reliance on annotated datasets, limited temporal modeling, insufficient occlusion reasoning, and a lack of structured object-level reasoning. By identifying these differences, this section clarifies how the proposed approach contributes novel insights and addresses limitations that persist in current methods.

3.1 Comparison with Key Papers

DGNN-YOLO: This framework [12] integrates a dynamic GNN with the YOLOv11 detector to track small objects under occlusion. In contrast, the proposed method is fully unsupervised and operates without any annotated data. DGNN-YOLO functions in 2D space and relies on supervised visual features, whereas the proposed system constructs 3D scene graphs from monocular RGB input using self-supervised depth estimation and unsupervised object discovery. The use of 3D geometry and occlusion-aware relational reasoning distinguishes the proposed approach from DGNN-YOLO's 2D spatial modeling.

Semi-Supervised Temporal GNN: While this method [15] leverages temporal GNNs and pseudo-labels to enhance 3D object detection in point cloud videos, it depends on a semi-supervised training loop with teacher-student supervision. The proposed method, in contrast, is fully unsupervised and operates on monocular RGB video, estimating depth and tracking objects through a joint GNN-segmentation loop. It also uses temporal embeddings to refine detection over time, unlike the fixed teacher-student update mechanism.

Batch3DMOT: This multimodal, offline tracking framework Batch3DMOT [1] utilizes camera, LiDAR, and radar data and relies on cross-modal attention and agglomerative clustering. Unlike the proposed framework, Batch3DMOT requires labeled trajectory annotations and is limited by its offline nature. The proposed model instead processes monocular video without supervision, using scene graphs and temporal consistency for tracking.

Iterative Scene Graph Generation: This work [6] focuses on generating scene graphs from single images using transformer-based iterative refinement. It does not handle temporal dynamics, tracking, or occlusion. In contrast, the proposed framework generates a sequence of 3D scene graphs and processes them with a GNN to track object identity over time. The iterative temporal modeling is an improvement over static graph generation in the context of tracking.

Context-Aware CompositionalNets: This framework [13] is designed for robust detection under occlusion in 2D images. It employs supervised part-based voting and disentangled representations. The proposed method instead leverages occlusion-aware 3D geometry and operates in a fully unsupervised setting. It handles occlusion by reasoning over depth-based spatial relationships within the scene graph, rather than with learned part based voting.

3.2 Comparison with Additional Studies

Unsupervised and Self-Supervised Tracking Methods: The reviewed existing methods, SimpleReID [5], UDT [16], and PUL [17], learn 2D embeddings based on pseudo-labeling, appearance re-identification, or contrastive learning. Despite being effective, these frameworks assume that motion or appearance remains consistent over time and lack structured spatial reasoning. In contrast, the proposed framework builds 3D scene graphs and models temporal relations using a GNN, capturing structure, motion, and occlusion in 3D space. It models tracking as a spatiotemporal reasoning problem rather than re-identification task for every frame, allowing it to handle dynamic occlusions more robustly.

Graph-Based Tracking (GCNNMatch): GCNNMatch [11] is a supervised online method that computes edge affinities between detections and tracklets using a bipartite matching formulation. It requires precomputed detections and is trained with a binary cross-entropy loss on labeled associations. Unlike GCNNMatch, the proposed method eliminates detection supervision entirely and instead discovers object masks from raw video, tracks their evolution through GNN embeddings, and reasons about occlusion using explicit geometric edge features.

Occlusion Handling Mechanisms: The reviewed methods largely rely on supervised or offline-trained occlusion modules, where Zhang et al. [20] use occlusion-weighted GCNNs, and Liu et al. [9] rely on part-segmentation supervision. Wang et al. [14] segment dynamic and static regions using geometric constraints, but do not maintain persistent object identity or perform reasoning over time. Dong et al. [3] adaptively selects a classifier to best handle a scene depending on occlusion-level, but it does not incorporate temporal relationships. The proposed approach differs by representing occlusion directly in the 3D scene graph using relational cues like depth ordering and visibility overlap. It models occlusion temporally as a learned structural property in an online, self-supervised manner.

Dynamic 3D Scene Understanding: The reviewed works include methods for reconstructing 3D geometry, parsing semantic layouts, or identifying dynamic and static regions. Specifically, the aim of these papers is scene understanding rather than object tracking. The frameowork proposed by Lu et al. [10], as well as 3D-URN [2], PS-HM [4], U3DS^3 [8], and HMP3D [7], focus on parsing static 3D layouts or leveraging geometric properties in point clouds. Additionally, OpenFunGraph constructs scene graphs with pretrained VLMs and LLMs, but lacks identity tracking or temporal modeling. In contrast, the proposed framework uses 3D scene graphs as a representation to improve tracking robustness and occlusion awareness in a unified, unsupervised system. Unlike HMP3D [7] or PS-HM [4], it is trained end-to-end without labels, and unlike OpenFunGraph [19], it does not rely on pretrained foundation models.

Self-Supervised Representation Learning for GNNs: The reviewed method LaGraph [18], proposed by Yaochen Xie et al., demonstrates the effectiveness of self-supervised learning for GNNs. However, the design is primarily focused on static graphs and does not model edge dynamics or temporal evolution, both of which are essential for object tracking in video. Specifically, LaGraph focuses on learning node representations through feature masking and reconstruction, but it does not encode spatiotemporal relationships or reason about occlusion events. Additionally, the framework assumes a fixed graph structure during training, and therefore cannot be applied to dynamic scenes where object interactions and visibility change continuously. In contrast, the proposed framework applies self-supervised learning in a temporally evolving graph setting, where both node and edge features are used to learn robust object embeddings.

4 Methodology

This section outlines the proposed framework for unsupervised occlusion-aware 3D object tracking in video, which models object-level spatiotemporal relationships using 3D scene graphs and a Graph Neural Network (GNN). The framework combines geometry, object-centric segmentation, and spatiotemporal reasoning, and is designed to operate without any labelled data. It consists of four main modules: a self-supervised depth estimator, an unsupervised object detector, a 3D scene graph constructor, and a spatiotemporal GNN tracker. These components enable the framework to track objects across frames, reason about occlusion, and maintain persistent identity embeddings over time. Additionally, its modularity allows for joint training, and its reliance on self-supervision greatly improves scalability.

4.1 Depth Estimation Module

Geometric reasoning is crucial for robust object tracking, particularly in unsupervised settings where appearance features alone may fail under occlusion. To provide geometric cues to both the object detection and scene graph construction modules, a self-supervised depth estimation network was trained alongside a pose estimation network. The depth maps produced by this module support depth-aware object detection, while the resulting 3D positional information is incorporated into the node embeddings of the evolving scene graph.

The input video is first processed by a module inspired by the Monodepth2 architecture, which includes a depth prediction network and an auxiliary pose estimation network for adjacent frame pairs. For each frame, the model outputs a dense per-pixel depth map and estimates relative camera poses. These outputs supply geometric structure that complements appearance features: depth information improves segmentation by aligning boundaries with geometric discontinuities, while 3D positional cues enhance the scene graph representation used by the Graph Neural Network for spatiotemporal reasoning.

4.1.1 Depth Network Architecture

The depth estimation model follows an encoder–decoder architecture with skip connections. The encoder uses a ResNet-18 backbone pretrained on ImageNet, truncated before the final classification layers, to extract hierarchical image features. Intermediate feature maps are returned as skip connections to allow for high-resolution depth reconstruction. The decoder progressively upsamples the encoded features using transposed convolutions and fuses them with the corresponding skip features. The final depth prediction is produced via a 1×1 convolution, followed by bilinear upsampling to the input resolution. This design allows for the network to have a better understanding of the image as the image encoding is robust, and the decoder is able to reconstruct guided by reliable intermediate encoded features.

4.1.2 Pose Network Architecture

In order to estimate relative camera motion between consecutive frames, a simple pose network is implemented. The network takes a pair of images concatenated along the channel dimension as input and predicts six parameters that correspond to 3D rotation and translation. It consists of a sequence of convolutional layers with progressively increasing feature dimensions, followed by global average pooling and a linear projection. The predicted pose is converted into a transformation matrix, which is then used to warp source images into the target frame's viewpoint.

4.1.3 Training Objectives

The networks are trained jointly in a self-supervised manner using photometric reconstruction losses. Given a target image and the previous frame, the depth network predicts a disparity map, while the pose network estimates relative camera motion. The source image is then projected into the target view using the predicted depth and pose to construct a warped image. The difference between the warped and true target images is minimized. An additional edge smoothness loss is also implemented to guide image boundaries. The total loss is taken as a weighted sum of photometric reconstruction losses and the edge smoothness loss.

The training objective can be summarized as follows:

1. **Photometric loss** ($\alpha \times SSIM + (\alpha - 1) \times L1$):
   A weighted combination of structural similarity (SSIM) and pixel-wise L1 loss is used to encourage reliable results from the depth network.
2. **Photometric loss (L1)**:
   A reconstruction loss computed using only the L1 component to encourage stable training in the pose network.
3. **Edge-aware smoothness loss**:
   An image-gradient-weighted smoothness term regularizes the disparity map, thus promoting piecewise smooth depth predictions while preserving sharp object boundaries.

Training Strategy
To ensure stable convergence for both networks, training proceeds in stages. In the first stage, a primary depth network is trained jointly with an arbitrary pose network, using the photometric SSIM and L1 loss. In the next stage, a primary pose network is trained with some arbitrary depth network, using the photometric L1 loss. At this point, both primary networks have been adequately trained, and in the third stage, they can be jointly fine-tuned for a smaller amount of epochs to further improve performance. For the first quarter of the fine-tuning epochs, the pose network is frozen, which encourages the depth network to improve while having reliable reconstructions to work with. After the first quarter, the pose network is unfrozen and it is jointly fine-tuned with the depth network. Throughout the fine-tuning epochs, the contribution of the smoothness loss is gradually decayed over time, allowing for the model to prioritize reconstruction.

4.1.4 Previous Depth Module Experiments

Before implementing on the current training strategy for depth/pose estimation, two architectures and several loss combinations were explored. These experiments helped identify key issues in the joint learning of depth and pose networks, specifically regarding balancing loss functions to prevent network collapse. The attached table summarizes the main attempts and observations.

These experiments revealed that the depth network is highly sensitive to the relative weighting of the photometric reconstruction losses. In particular, the L1 loss primarily encourages accurate image reconstruction, which favours pose learning, while the SSIM term guides the depth network to capture meaningful geometric structure. The final strategy, involving sequential pretraining and staged fine-tuning with selective freezing and decayed smoothness weights, was adopted to mitigate collapse and improve depth quality while maintaining reliable pose estimates.

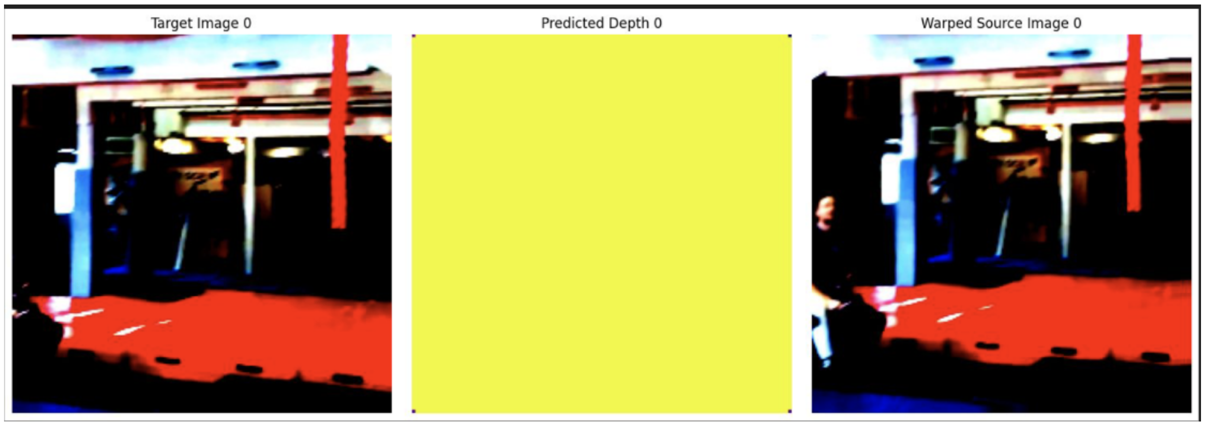

Figure 2: Skip Connection Depth with L1 loss

Architecture / Loss	**Observations**
Simple ResNet encoder, no skip connections, upsample + Conv2D decoder	Learning rate sensitivity: lr $< 1e^{-4}$ oscillated, lr $> 1e^{-5}$ plateaued around 0.79. Depth maps extremely vague with poor structure, even at seemingly ideal lr $= 1e^{-4}$.
ResNet encoder with skip connections, ConvTranspose upsampling decoder	• L1 photometric loss: depth maps flat, warped image accurate $\Rightarrow$ pose learned well. • SSIM + L1 loss: depth maps accurate, warped image collapsed $\Rightarrow$ pose failed. • Loss balancing analysis: L1 encourages pose accuracy, SSIM encourages depth learning; simultaneous training causes depth to rely entirely on pose. • Sequential training: primary depth with arbitrary pose (SSIM + L1), primary pose with arbitrary depth (L1), then fine-tuned jointly. • Freeze pose for early fine-tuning epochs, then unfreeze to allow depth improvement. Gradual decay of edge-smoothness weight prioritized reconstruction loss.

Table 1: Summary of preliminary depth module experiments.

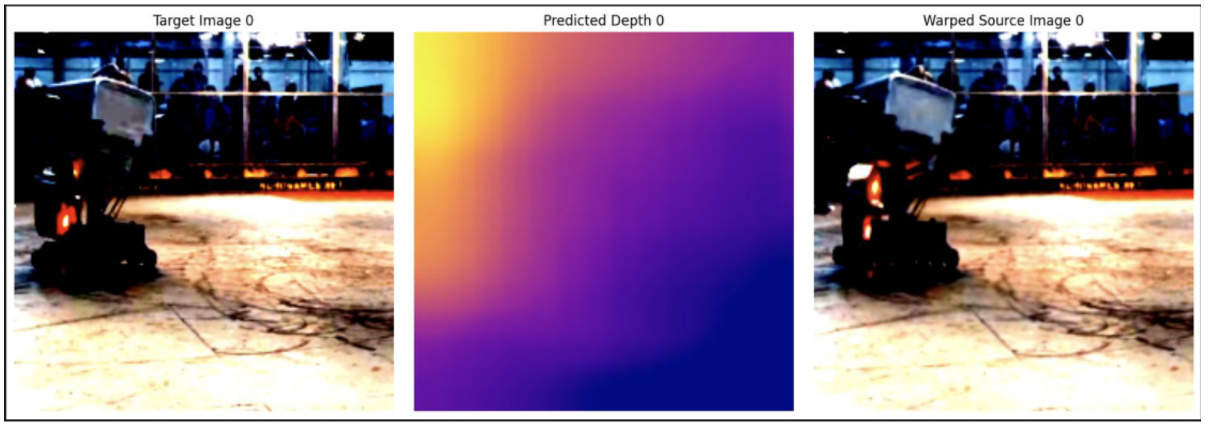

Figure 1: Simple Depth with lr=1e-4

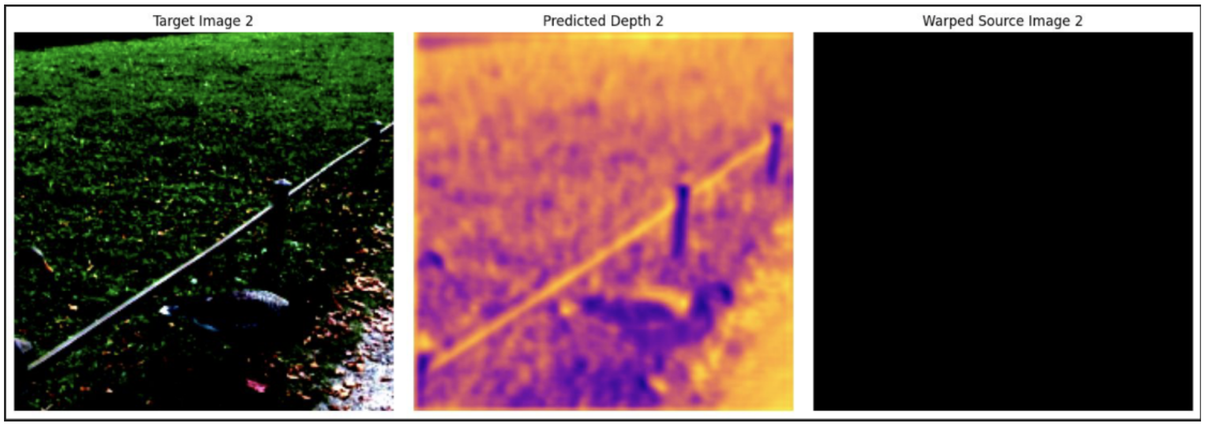

Figure 3: Skip Connection Depth with SSIM+L1 loss

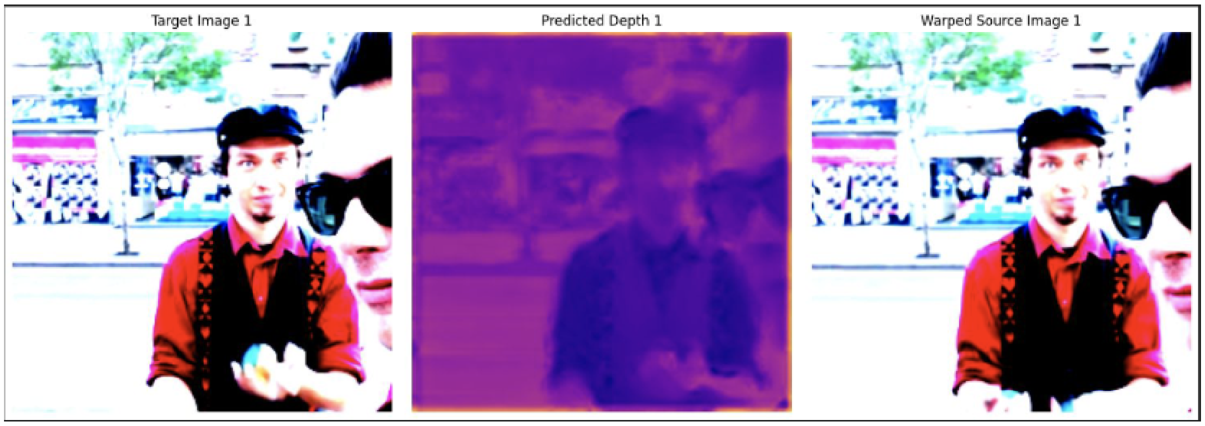

Figure 4: Pretrained + Finetuned

4.2 Object Detection Module

Object detection in this framework is entirely unsupervised and leverages depth cues from the depth estimation module to improve segmentation quality. Rather than relying on pre-trained class-specific detectors, the system adopts a slot-based object representation, where each slot corresponds to a candidate object in the scene. This allows the model to generalize across categories and robustly segment objects without explicit ground truth annotations. The module jointly reasons over appearance and geometric information by integrating 2D feature maps from the visual backbone model (DINO-based) with a per-pixel depth map from the pre-trained depth network. The resulting object representations are then passed to the scene graph construction and GNN components of the pipeline.

4.2.1 Feature Extraction

Each input frame is initially encoded by a DINO-based vision transformer backbone, which produces dense feature embeddings. Additionally, the frame is processed by the depth estimation network to generate a depth map, which is bilinearly downsampled to match the spatial resolution of the feature map ($16x16$). These two values are concatenated to form a depth-augmented feature map that provides both appearance and geometric cues for object detection.

4.2.2 Slot Attention Mechanism

To segment the scene into discrete entities, a Slot Attention module is employed with a fixed number of slots. Each slot is initialized with a Gaussian prior and iteratively updated through cross-attention with the feature map. At each iteration, queries derived from the current slot embeddings attend to keys and values derived from the depth-augmented feature map. Attention weights act as soft segmentation masks, assigning pixels to slots in a competitive manner. The slot embeddings are refined via a recurrent GRU cell and an MLP residual update, enabling them to capture coherent object-level features across multiple refinement steps. This iterative process yields a set of slot vectors, each representing a candidate object hypothesis.

4.2.3 Mask Decoding and Segmentation

The slot embeddings are then decoded into spatial object masks using a simple Mask Decoder. Each slot is projected back into the feature space, and attention is computed against the feature map to construct slot-specific masks. These masks serve as soft detections, which are reshaped into spatial maps that emphasize the pixels corresponding to each object candidate.

4.2.4 Object Features and Bounding Boxes

In order to obtain explicit object-level descriptors, the module computes object features as masked averages of the feature and depth maps by pooling. The feature vector associated with each slot thus contains both appearance and geometric information, capturing not only visual context but also 3D positional cues. Bounding boxes are derived directly from the spatial masks by computing the minimum enclosing rectangle of nonzero pixels above a threshold. A confidence score is assigned to each detection, based on a weighted combination of the average and peak mask activations as well as a threshold on bounding box minimum area. These confidence values are later used to prune additional slots.

4.2.5 Integration with Scene Graph

The resulting object embeddings, masks, and bounding boxes form the set of detected objects for each frame. These outputs are passed to the scene graph construction module, where objects are treated as nodes and their relations are inferred through spatial and temporal reasoning. By relying on unsupervised slot-based segmentation enriched with depth cues, the detection module avoids the limitations of class-specific detectors while still producing structured, object-centric representations suitable for downstream reasoning tasks.

4.3 3D Scene Graph Construction Module

The scene graph construction module is responsible for representing the detected objects and their relationships in a structured, graph-based form. Each frame is mapped into a scene graph, where nodes correspond to objects and edges capture their pairwise relationships. The design incorporates both spatial and temporal information to model object interactions in dynamic scenes. This step is constructed such that it is differentiable and allows for backpropogation.

4.3.1 Scene Graph Constructor

The SceneGraphConstructor first pools appearance and geometric features for each detected object as follows:

- Node features are constructed by combining multiple features: pooled visual features from the backbone, object-specific embeddings from the detection module, bounding box positional encodings, depth-informed motion priors using the depth and pose networks, and detection confidence scores. These are passed through a multi-layer perceptron (MLP) to produce compact node embeddings.
- Edge features are computed for each pair of nodes by concatenating their embeddings, which are then processed through an edge MLP. This produces edge embeddings capturing potential object relations.
- A soft adjacency matrix is computed based on embedding similarity to encode possible relations between objects.

The result is a fully connected scene graph representation for each frame, parameterized by node and edge embeddings along with their adjacency structure.

4.3.2 Temporal Evolution

While static scene graphs capture intra-frame relationships, video data requires temporal consistency. To this end, the EvolvingSceneGraphTemporal model is implemented, which maintains and updates the scene graph across frames.

- Node embeddings are updated recurrently using a GRU, ensuring that object representations evolve smoothly as they move across time.
- Temporal edge embeddings are introduced to model cross-frame relationships. These are computed by comparing node embeddings from the previous and current frames, then processed via an additional MLP. This enables the system to reason about object trajectories and temporal interactions.
- At each timestep, the module outputs updated node embeddings, intra-frame edge embeddings, temporal edge embeddings, and an adjacency matrix.

This evolving scene graph formulation allows the system to maintain temporal coherence, handle partial occlusions, and capture dynamic relationships, which are essential for tasks such as video reasoning, interaction prediction, and long-term scene understanding.

4.4 Spatiotemporal GNN Tracker

This module is a graph neural network designed to perform relational reasoning over scene graphs and produce identity-preserving node embeddings. It refines these embeddings and determines which cues are most informative for maintaining object identity across frames. This essentially defines the temporal component of object tracking.

4.4.1 Node and Edge Embedding Projections

The GNN Tracker first projects node embeddings, intra-frame edge embeddings, and temporal edge embeddings into a common latent space using learnable linear transformations. This ensures that appearance, geometry, and motion cues can be jointly processed. Temporal edge features are integrated into the intra-frame edge features, enabling the model to explicitly encode motion consistency and cross-frame relationships.
These learning linear transformations are as follows:

$$\mathbf{h}_i = W_n \mathbf{x}_i, \quad \mathbf{e}_{ij} = W_e \mathbf{z}_{ij}, \quad \mathbf{e}_{ij}^{temp} = W_t \mathbf{z}_{ij}^{temp},$$

where $\mathbf{x}_i \in \mathbb{R}^{d_n}$ is the embedding of node i, $\mathbf{z}_{ij} \in \mathbb{R}^{d_e}$ is the intra-frame edge embedding between nodes i and j, and $\mathbf{z}_{ij}^{temp}$ is the temporal edge embedding across frames. The projected temporal edge is combined with the spatial edge as:

$$\mathbf{e}_{ij} \leftarrow \mathbf{e}_{ij} + \mathbf{e}_{ij}^{temp}$$

4.4.2 Message Passing and Aggregation

To propagate contextual information, the GNN performs message passing between nodes. For each object node, messages are constructed by taking the element-wise product of its embedding with the edge features connecting it to other objects. This allows each node to dynamically weigh different relational cues depending on the scene context such as relying more on motion priors than appearance features when dealing with occlusion. Aggregated messages from all neighbours are summed, producing an updated feature representation for each node.
For a node i, we define the message from node j as the element-wise product of the projected edge and node embeddings:

$$\mathbf{m}_{ij} = \mathbf{e}_{ij} \odot \mathbf{h}_j,$$

where $\odot$ denotes element-wise multiplication. The aggregated message for node i is then obtained by summing over all neighbours:

$$\mathbf{m}_i = \textstyle\sum_{j \neq i} \mathbf{m}_{ij}.$$

4.4.3 Identity-Preserving Updates

The refined messages are processed by a GRU-based update function, which maintains a hidden state for each node. This recurrent mechanism provides temporal continuity, which ensures that the updated node embeddings evolve smoothly across frames. By incorporating both past states and new relational information, the GRU enables the model to preserve long-term object identity even under challenging conditions such as occlusion.
For each node i, the update step is:

$$\mathbf{h}'_i = \mathrm{GRU}(\mathbf{m}_i, \mathbf{h}_i),$$

where $\mathbf{h}'_i$ denotes the refined node embedding, $\mathbf{m}_i$ is the aggregated message, and $\mathbf{h}_i$ is the previous hidden state.

4.4.4 Output Representations

The GNN Tracker outputs:

- Refined node embeddings $\mathbf{h}'_i$ that preserve object identity across frames.
- Updated edge embeddings $\mathbf{e}_{ij}$ that integrate spatial and temporal relationships.

These refined representations form the foundation for unsupervised object tracking, enabling the framework to associate objects across frames without ground truth annotations.

4.5 Training Strategies

To evaluate the contributions of temporal information and relational reasoning, we consider two training strategies:

- **object detection independent training**: where the object detection module is trained independently to evaluate how a model without temporal understanding, relying purely on spacial understanding, would perform.
- **joint training**, where the object detection, scene graph construction, and GNN tracker are trained together, to compare the effects of having a temporal understanding as well as a spatial understanding.

4.5.1 Object Detection Independent Training

In this stage, only the **Object Detector** is trained in an unsupervised manner. The training objective combines several self-supervised losses designed to encourage meaningful object segmentation:

1. **Mask Reconstruction Loss**:

$$\mathcal{L}_{recon} = \tfrac{1}{BCHW} \textstyle\sum_{b,c,h,w} \left(\hat{F}_{b,c,h,w} - F_{b,c,h,w}\right)^2,$$

   where $\hat{F}$ is the reconstructed feature map obtained from slot features and masks, and F is the original backbone feature map.

2. **Mask Entropy Loss**:

$$\mathcal{L}_{ent} = -\tfrac{1}{BSHW} \textstyle\sum_{b,s,hw} \left(m_{b,s,hw} \log m_{b,s,hw} + (1 - m_{b,s,hw}) \log(1 - m_{b,s,hw})\right)$$

   which encourages masks to be confident and near-binary.

3. **Depth Edge Alignment Loss**:

$$\mathcal{L}_{depth} = \tfrac{1}{BSHW} \textstyle\sum_{b,s,h,w} m_{b,s,h,w} \left\|\nabla D_{b,h,w}\right\|,$$

   aligning masks with geometric discontinuities captured by the predicted depth map D.

4. **Mask Orthogonality Loss**:

$$\mathcal{L}_{ortho} = \tfrac{1}{B} \textstyle\sum_{b} \tfrac{1}{S(S-1)} \textstyle\sum_{i \neq j} m_i^\top m_j,$$

   encouraging masks to be distinct and non-overlapping.

The total object detection loss is a weighted sum:

$$\mathcal{L}_{obj} = \lambda_{recon}\mathcal{L}_{recon} + \lambda_{ent}\mathcal{L}_{ent} + \lambda_{ortho}\mathcal{L}_{ortho} + \lambda_{depth}\mathcal{L}_{depth}.$$

4.5.2 Joint Training of Detection, Scene Graph, and GNN

In the second strategy, the object detection, scene graph construction, and GNN tracker are trained jointly. The GNN refines node embeddings across frames, and the network is trained using a **contrastive loss** that encourages temporal consistency:

$$\mathcal{L}_{contrast} = -\tfrac{1}{BN} \textstyle\sum_{b,i} \log \tfrac{\exp(\mathbf{h}_i^{(t)} \cdot \mathbf{h}_i^{(t+1)}/\tau)}{\sum_j \exp(\mathbf{h}_i^{(t)} \cdot \mathbf{h}_j^{(t+1)}/\tau)},$$

where $\mathbf{h}_i^{(t)}$ and $\mathbf{h}_i^{(t+1)}$ are normalized node embeddings at consecutive frames, N is the number of nodes, and τ is a temperature hyperparameter.

During training, each video sequence is processed sequentially:

1. The object detector produces feature maps, masks, and object embeddings.
2. The evolving scene graph constructor generates node and edge embeddings, incorporating spatial cues and motion priors.
3. The GNN tracker refines the node embeddings based on spatial and temporal relations.
4. Contrastive loss is applied between consecutive node embeddings to enforce temporal consistency.

This joint training strategy allows the model to leverage both spatial and temporal cues, improving object segmentation and identity preservation across frames. It also provides a way to evaluate how temporal reasoning enhances unsupervised object detection performance.

5 Analysis of Experimental Results

5.1 Object Detection Individual Training

After training the object detection module in isolation for 20 epochs, a failure mode was observed where the predicted bounding boxes appearing in the same region of the image. This indicates a form of model collapse, where the slot attention mechanism and mask decoder fail to learn distinct object representations. There are several factors that may be contributing to this behaviour: the model may be overfitting to a small subset of features, or the gradients from the reconstruction and regularization losses may have vanished or have become too weak to effectively guide learning. Despite incorporating the depth cues from the pre-trained depth network, the object detector did not appear to utilize this information effectively under these training conditions.

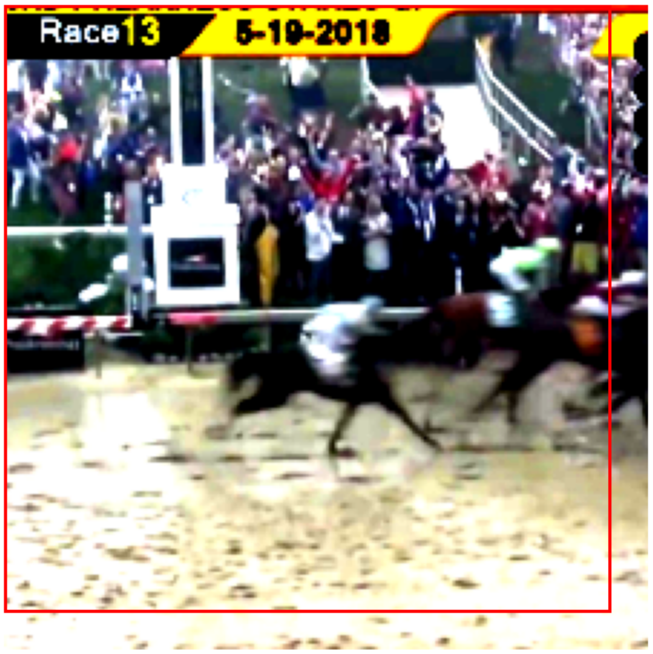

Figure 5: Result from individual object detection training

5.2 Joint Training of Detection, Scene Graph, and GNN

When training the object detector from scratch jointly with the evolving scene graph constructor and GNN tracker, the predictions improved in some respects but remained poor. The resulting bounding boxes were noisy yet seemed slightly more temporally consistent across consecutive frames compared to the individual object setup. This improvement suggests that the GNN and scene graph modules are capable of propagating relational and temporal information, which can provide a stabilizing influence on the embeddings. However, the overall performance remained poor, most likely because the object detector was completely untrained. As a result, the initial noise in the detection masks and features propagated through the scene graph and GNN, destabilizing training and preventing the model from learning reliable object representations.

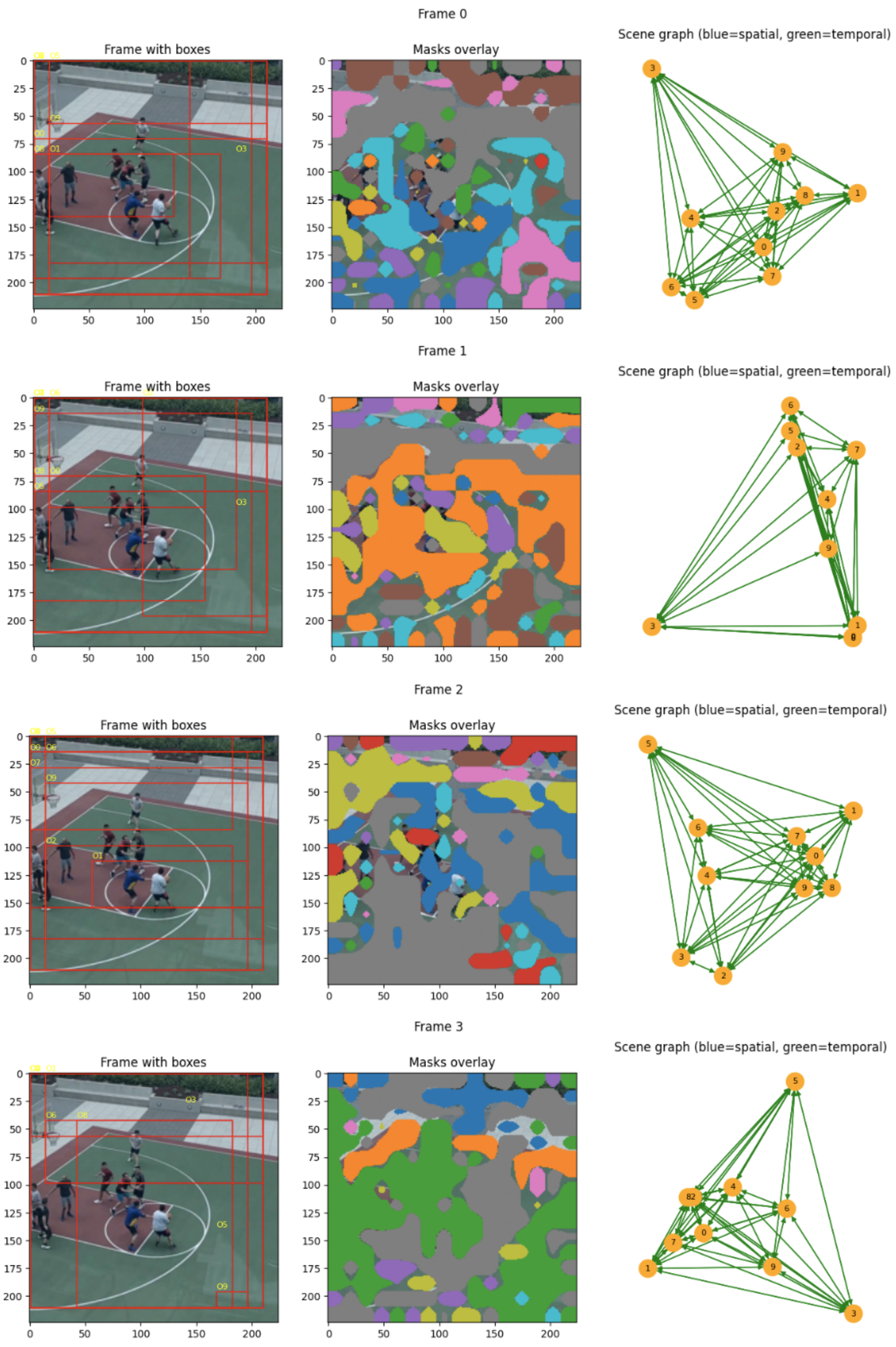

Figure 6: Result from GNN-object joint training

5.3 Summary

Rather than displaying an observable difference in spatial-only vs. spatiotemporal understanding, the experiments seem to demonstrate the importance of a stable object detector in the object tracking task. While temporal and relational reasoning may be able to partially mitigate the poor object representations, they cannot compensate for an untrained or collapsing detection module.

6 Conclusion

This research project explored a novel framework for unsupervised, occlusion-aware object tracking by integrating self-supervised depth estimation, unsupervised object detection, 3D evolving scene graph construction, and a spatiotemporal Graph Neural Network. The central motivation was to address prevalent limitations in existing approaches, including reliance on annotated data, lack of persistent object identity under occlusion, and limited temporal modelling beyond short sequences.

A series of experimental attempts with the depth estimation module revealed critical insights into the interplay between depth and pose learning. The first design used a simple encoder-decoder architecture with basic upsampling and convolving layers, which produced vague results. The second architecture with skip connections and learnable upsampling, demonstrated the competing objectives of L_1 and SSIM-based photometric losses. By training the depth and pose networks in two rounds, freezing and unfreezing pose, and employing a decaying smoothness weight during fine tuning, the module achieved improved structural understanding in depth maps while maintaining robust pose predictions. These design choices proved to be useful in ensuring stable learning dynamics for this module.

The object detection module presented more significant challenges. After only 20 epochs of training, bounding box predictions appeared to collapse, appearing only in the same single area of the frame. This collapse may have been due to overfitting, gradient vanishing, or convergence at a poor local minimum due to a small learning rate. When trained jointly with the GNN tracker in an end-to-end manner, results remained noisy. Although bounding boxes exhibited some minimal degree of temporal consistency, the untrained detection module likely propagated noise throughout the system, destabilizing the overall training process.Despite these challenges, the project demonstrates the feasibility of integrating depth estimation, object detection, scene graph construction and temporal graph reasoning into an unsupervised end-to-end framework.

In conclusion, this project highlights the practical difficulties of building a fully unsupervised framework for occlusion-aware tracking. While current results remain preliminary and limited by unstable detection performance, the insights gained provide grounds for further investigation. Future work should explore integrating a pretrained depth estimation network, as well as a pretrained object detection module that can provide stable initial object detections. This can allow for the differences in spatiotemporal vs. spatial models to be more evident. Additionally, scene graph embeddings can be further refined by implementing occlusion-reasoning with geometric constraints. Additionally, the model can be trained with a dataset that contains longer sequence of frames. However this can increase the amount of time take during training significantly. These extensions could improve the framework, which would ultimately advance progress towards occlusion aware object tracking in real-world environments.

References

[1] Martin Buchner and Abhinav Valada. *3D Multi-Object Tracking Using Graph Neural Networks with Cross-Edge Modality Attention*. 2022. arXiv: 2203.10926 [cs.CV]. URL: https://arxiv.org/abs/2203.10926.

[2] Geonho Cha, Minsik Lee, and Songhwai Oh. "Unsupervised 3D Reconstruction Networks". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[3] Xingping Dong et al. "Occlusion-Aware Real-Time Object Tracking". In: *IEEE Transactions on Multimedia* PP (Nov. 2016), pp. 1–1. DOI: 10.1109/TMM.2016.2631884.

[4] Vitor Guizilini and Fabio Ramos. "Unsupervised Feature Learning for 3D Scene Reconstruction with Occupancy Maps". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 31.1 (2017). DOI: 10.1609/aaai.v31i1.11039. URL: https://ojs.aaai.org/index.php/AAAI/article/view/11039.

[5] Shyamgopal Karthik, Ameya Prabhu, and Vineet Gandhi. *Simple Unsupervised Multi-Object Tracking*. 2020. arXiv: 2006.02609 [cs.CV]. URL: https://arxiv.org/abs/2006.02609.

[6] Siddhesh Khandelwal and Leonid Sigal. *Iterative Scene Graph Generation*. 2022. arXiv: 2207.13440 [cs.CV]. URL: https://arxiv.org/abs/2207.13440.

[7] Kevin Lai, Liefeng Bo, and Dieter Fox. "Unsupervised feature learning for 3D scene labeling". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 3050–3057. DOI: 10.1109/ICRA.2014.6907298.

[8] Jiaxu Liu et al. "U3DS3: Unsupervised 3D Semantic Scene Segmentation". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2024, pp. 3759–3768.

[9] Qiankun Liu et al. *Online Multi-Object Tracking with Unsupervised Re-Identification Learning and Occlusion Estimation*. 2022. arXiv: 2201.01297 [cs.CV]. URL: https://arxiv.org/abs/2201.01297.

[10] Feixiang Lu et al. *Real-time 3D scene reconstruction with dynamically moving object using a single depth camera*. 2018. DOI: https://doi.org/10.1007/s00371-018-1540-8.

[11] Ioannis Papakis, Abhijit Sarkar, and Anuj Karpatne. *GCNNMatch: Graph Convolutional Neural Networks for Multi-Object Tracking via Sinkhorn Normalization*. 2021. arXiv: 2010.00067 [cs.CV]. URL: https://arxiv.org/abs/2010.00067.

[12] Shahriar Soudeep, Md Abrar Jahin, and M. F. Mridha. *Interpretable Dynamic Graph Neural Networks for Small Occluded Object Detection and Tracking*. 2025. arXiv: 2411.17251 [cs.CV]. URL: https://arxiv.org/abs/2411.17251.

[13] Angtian Wang et al. *Robust Object Detection under Occlusion with Context-Aware CompositionalNets*. 2020. arXiv: 2005.11643 [cs.CV]. URL: https://arxiv.org/abs/2005.11643.

[14] Guangming Wang et al. "Unsupervised Learning of Depth, Optical Flow and Pose With Occlusion From 3D Geometry". In: *IEEE Transactions on Intelligent Transportation Systems* 23.1 (2022), pp. 308–320. DOI: 10.1109/TITS.2020.3010418.

[15] Jianren Wang et al. *Semi-supervised 3D Object Detection via Temporal Graph Neural Networks*. 2023. arXiv: 2202.00182 [cs.CV]. URL: https://arxiv.org/abs/2202.00182.

[16] Ning Wang et al. *Unsupervised Deep Tracking*. 2019. arXiv: 1904.01828 [cs.CV]. URL: https://arxiv.org/abs/1904.01828.

[17] Qiangqiang Wu, Jia Wan, and Antoni B. Chan. *Progressive Unsupervised Learning for Visual Object Tracking*. 2021. URL: https://openaccess.thecvf.com/content/CVPR2021/html/Wu_Progressive_Unsupervised_Learning_for_Visual_Object_Tracking_CVPR_2021_paper.html.

[18] Yaochen Xie, Zhao Xu, and Shuiwang Ji. *Self-Supervised Representation Learning via Latent Graph Prediction*. 2022. arXiv: 2202.08333 [cs.LG]. URL: https://arxiv.org/abs/2202.08333.

[19] Chenyangguang Zhang et al. *Open-Vocabulary Functional 3D Scene Graphs for Real-World Indoor Spaces*. 2025. arXiv: 2503.19199 [cs.CV]. URL: https://arxiv.org/abs/2503.19199.

[20] Yubo Zhang, Liying Zheng, and Qingming Huang. "Occlusion-related graph convolutional neural network for multi-object tracking". In: *Image and Vision Computing* 152 (2024), p. 105317. ISSN: 0262-8856. DOI: https://doi.org/10.1016/j.imavis.2024.105317. URL: https://www.sciencedirect.com/science/article/pii/S0262885624004220.