A Modular Architecture to Enhance Object Permanence in Object Tracking

Ekin Kağan Özkan
Defence Technologies Master Program
Istanbul Technical University
ozkan23@itu.edu.tr
514231009

I. Abstract

Object permanence, the ability to predict object locations when not visible, is a fundamental cognitive skill and a challenging task in computer vision, particularly in scenarios involving occlusions and containment. This paper presents a novel modular pipeline for addressing object permanence, breaking the task into interconnected components: object tracking, lost object detection, depth extraction, video classification, and location prediction. The modular design allows flexibility, adaptability, and continuous improvement. Evaluated on the CATER dataset, the pipeline achieved 80% overall accuracy in predicting object locations during occlusion and containment. Additionally, the impact of monocular depth information was explored through two new datasets, CATER-Action and CATER-Depth. Results highlight the pipeline's robustness and flexibility in tackling object permanence challenges.

II. Introduction

Object tracking faces challenges when objects become temporarily invisible, requiring systems to handle non-visible scenarios similar to the human concept of object permanence, first defined by Piaget (1954) [8]. Shamsian et al. [3] categorized these scenarios into visible, occluded, contained, and carried. While "visible" is straightforward, "occluded" requires predicting locations based on partial or past data, and "contained" scenarios demand reasoning about the container's movements.

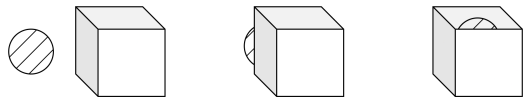

Fig. 1: Visualization of the inspected cases

Our work focuses on classifying "occluded" and "contained" cases using a flexible pipeline with time-series video data. We explore two approaches: one using original images and the other leveraging monocular depth estimation. To support this, we introduce two datasets, CATER-Action and CATER-Depth. Key contributions include developing an adaptable pipeline, assessing monocular depth for object permanence, and creating new datasets to aid future research.

III. Related Work

Object permanence is crucial in object tracking, especially for occluded or non-visible objects. Tokmakov et al. (2021) [1] extended CenterTrack with memory modules to track occluded objects, achieving state-of-the-art results on benchmarks. Traub et al. (2024) [2] introduced Loci-Looped, an unsupervised model using temporal imaginations for better occlusion handling. Shamsian et al. (2020) [3] proposed OPNet, excelling in scenarios like containment and carrying on the CATER dataset. These advancements emphasize spatio-temporal reasoning and memory for robust tracking in occlusion-rich environments.

IV. Methodology

A. Background

Inspired by Shamsian et al. [3] and their categorization of non-visible scenarios—visible, occluded, contained, and carried—we developed a modular pipeline for the object permanence task. Our method incorporates a monocular depth extraction module to improve reasoning about object locations in complex scenarios. Similar to Tokmakov et al. [1], which enhances CenterTrack with a spatio-temporal memory module, our pipeline focuses explicitly on addressing the 2 non-visible scenarios, leveraging depth and temporal reasoning for improved robustness. We demonstrate its effectiveness through evaluations on the CATER dataset.

B. Pipeline

The pipeline proposed in this paper addresses the object permanence task by breaking it into a series of connected and modular steps, ensuring flexibility, adaptability, and ease of improvement.

We propose a modular pipeline for the object permanence task, ensuring flexibility and adaptability. The process begins with object tracking to monitor and localize objects across frames, identifying trajectories and visibility states. Lost objects—those visible in the previous frame but missing in the current—are detected, and relevant footage is extracted. Monocular depth estimation is then used to enhance spatial

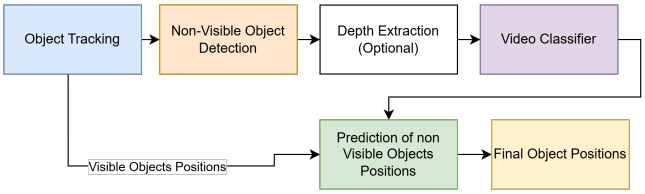

Fig. 2: Flowchart of the constructed pipeline

understanding. In the classification step, models like VideoMAE determine whether the object is occluded, contained, or a false positive. For occlusions, the object's location is assumed to be its last known position, while for containment, it is tied to the container until reappearance. The predicted location is then integrated with tracking results for coherent outputs. This modular design ensures scalability and easy adaptability to various scenarios and datasets.

C. Object Tracking

For the object tracking task, we integrated the DeepSort algorithm into our pipeline. While DeepSort has been surpassed by more recent methods such as ByteTrack [9], we chose DeepSort due to its active maintenance and updates. To enhance its tracking performance, we implemented several modifications and improvements within our pipeline.

DeepSort relies on an object detection model to perform tracking. Initially, we trained an object detection model using the CATER Object dataset, which we specifically curated for this task. However, the results did not meet our expectations in terms of accuracy. Consequently, we adopted the pre-trained object detection model from the work of Shamsian et al.[3], which significantly improved the tracking performance for our task.

To enhance the robustness of traditional object tracking pipelines, we incorporated several additional mechanisms beyond standard DeepSORT, focusing on data association improvements and handling the challenge of lost objects. These contributions aim to address common tracking difficulties, such as occlusions and object disappearance during a video sequence.

1) Object Dictionary: The object dictionary is a Python dictionary (objectDictionary) where each key corresponds to a unique identifier (ID) for a detected object, and the value is a nested structure that contains the following fields:
ID: The key in the dictionary represents the globally unique ID of the tracked object. This ID remains consistent during the object's lifetime in the video.
Bounding Box (bbox): The dictionary stores the latest bounding box coordinates for the object, which depicts the region of the image where the object is located.
Visual Embedding (embedding): A visual feature vector (embedding) generated by ResNet-50 is extracted for the object's cropped bounding box. This embedding encodes the object's appearance and is normalized to unit length to enable similarity calculations. It plays a crucial role in re-identifying objects once they are lost or occluded.
Label (label): The dictionary stores the predicted class label for the object (e.g., vehicles, pedestrians, animals, etc.), which is obtained from the Faster R-CNN object detection model.
Reputation Score (reputation): A reputation score associated with the object reflects its persistence in the tracking process. Reputation increases as the object is successfully detected across frames, and it resets if the object becomes "lost."

2) Reputation-Based Tracking: We implemented a reputation mechanism to evaluate the reliability of tracked objects over time. Each object is assigned a reputation score that increases with consecutive detections. Objects with scores above a specified threshold are marked as "reliable," distinguishing consistently detected objects from transient ones caused by noise or false positives.

3) Handling Lost Objects: To address scenarios where objects disappear (e.g., due to occlusion or tracking failures), we added a lost object handling module. It flags objects as "lost" if they are missing in successive frames but had high reputation scores, storing their ID, last bounding box, and frame. Additionally, we buffer pre and post-disappearance frames using a circular queue to create cropped video clips around the object's bounding box, with spatial padding. This feature was used to generate a dataset for model training.

D. Depth Extraction

We utilized Depth Anything V2 [5] for monocular depth estimation, as it demonstrated superior performance compared to competing methods in one-shot depth estimation tasks. By extracting depth information, our goal was to enhance the understanding of the spatial relationships between objects and the actions occurring within the image.

E. Dataset

In our work, we utilized the CATER dataset [6], originally introduced by Girdhar and Ramanan as a diagnostic dataset for compositional actions and temporal reasoning. Building upon this dataset, we derived two new datasets tailored to our specific research objectives: CATER-Depth and CATER-Action. These datasets were designed to address distinct challenges in our study, enabling a more comprehensive analysis of depth-related and action-based tasks.

1) CATER-Action: The CATER-Action dataset was constructed using our object detection module and a modified version of DeepSort. From the original CATER dataset, we extracted video segments featuring object occlusion and object containment scenarios.

These segments included both local footages, focusing solely on the region where the object disappearance occurs, and full-window footages capturing the entire scene. Subsequently, we annotated 1458 videos with labels such as

Class	Number of Videos
Occluded	863
Contained	147
False	448

TABLE I: Dataset Composition

"contained," "occluded," and "false" to categorize the observed interactions.

This labeled dataset was then used to train our real video classifier, enabling it to effectively recognize and classify these specific actions.

2) CATER-Depth: To enhance the classification performance of our model, we incorporated monocular depth extraction into our pipeline using the Depth Anything V2 model. This state-of-the-art depth estimation framework was applied to the CATER dataset, enabling us to extract detailed depth information from the original video sequences. By integrating depth data alongside the original frames, we aimed to provide our model with a richer understanding of the spatial relationships and geometric structure within the scenes.

F. Video Classifier

For the video classification task, we employed a model based on the VideoMAE[10] architecture (Masked Autoencoders for Video) using the pretrained 'MCG-NJU/videomae-base' model [11] from Hugging Face Transformers. VideoMAE's ability to effectively capture spatiotemporal features through self-supervised learning made it well-suited for this task. We fine-tuned the model to classify short video clips into one of three classes: "occluded," "contained," and "false," aiming to distinguish these scenarios based on visual and contextual cues.

To assess the impact of depth information, we trained two models on different datasets: the original CATER-Action dataset and its augmented version, CATER-Action Depth, which includes depth maps generated by the Depth Anything V2 model.

G. Object Location Predictor

This module of our pipeline is responsible for predicting the possible location of lost objects in the video frames. It achieves this by leveraging the predictions generated during the video classification step. Based on the class label received from the video classifier—"occluded", "contained", or "false"—the module applies a set of predefined rules to infer the location of non-visible objects. These rules are designed to handle different scenarios as follows:

If an object is classified as "occluded", the module assumes that the object remains in its last known position until it reappears in the frame. This rule is based on the assumption that occluded objects are temporarily hidden but have not moved significantly from their previous location.

If an object is classified as "contained", the module ties the object's location to the container object. The contained object is assumed to move in conjunction with the container until it reappears or is released. This rule accounts for scenarios where objects are placed inside other objects and their movements are dependent on the container's trajectory.

V. Experiments

A. Object Tracking

To evaluate the effectiveness of our improvements to the DeepSort object tracking algorithm, we conducted a comparative analysis between the original DeepSort and our enhanced version. For this evaluation, we selected a subset of videos from our dataset, specifically chosen to include challenging scenarios such as occlusions, object interactions, and complex motion patterns. These videos were designed to test the robustness and accuracy of both tracking algorithms under conditions that closely resemble real-world challenges.

After running both versions of the algorithm on the selected videos, we measured their performance by calculating the duration of accurate object tracking in seconds by randomly selecting an object.

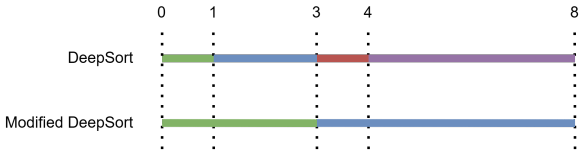

Fig. 3: This figure visualizes the performance of the two algorithms, with horizontal lines representing object tracking over time. Color changes indicate ID switches, and numbers mark the seconds when these changes occur. Fewer color changes reflect more stable tracking, demonstrating how our improved algorithm outperforms the original in maintaining object identity.

B. Video Classification

1) Comparing Depth and Real-Image Datasets: For the video classification task, we trained two distinct models to evaluate the impact of depth information on classification performance. The first model was trained using real-color RGB video data, while the second model was trained using monocular depth-extracted videos, where depth maps were generated using the Depth Anything V2 model. Both models were trained with 5 different randomized environments to decrease the randomness of the comparison results.

After training, we conducted a comprehensive evaluation to compare the performance of the two models. The results demonstrated that the depth-based resulted close results with the real-color model across multiple metrics, including accuracy, precision, and recall. While the depth-based model did not outperform the RGB model in all metrics and was sometimes worse in certain aspects, it demonstrated a closer performance to the RGB model and provided valuable insights into object interactions and spatial relationships. This suggests that depth information can serve as a useful feature for

enhancing classification tasks, particularly in scenarios where visual ambiguity or overlapping objects challenge traditional RGB-based models.

Dataset	Mean Accuracy	Mean F1 Score
RGB	0.9629±0.0074	0.9632±0.0066
Depth	0.9356±0.0099	0.9338±0.0094

TABLE II: Comparison of Results for Real Image Dataset and Depth Dataset

These findings highlight the potential of incorporating depth information as a supplementary feature in video classification tasks, even if it does not consistently surpass RGB-based models. The depth-based model's ability to contribute to spatial reasoning and object localization underscores its utility in applications where such information is critical, despite its limitations in some performance metrics.

2) Experimenting with Classes: To validate the robustness of our video classifier approach, we introduced a false dataset consisting of videos that depict scenarios such as partial occlusions, no-action sequences, and stationary objects. These videos were specifically designed to challenge the classifier by presenting cases where objects are not actively interacting or are only partially visible.

Despite these complexities, our video classifier models demonstrated acceptable performance on the false dataset, correctly identifying and categorizing the majority of cases. While the contained results were close in both 3 class and 2 class datasets, occluded results had a difference, the major cause of this situation is that the false class contained partly occluded results too. This results confirms the classifier's ability to handle ambiguous or non-interactive scenarios, highlighting its robustness and generalizability across diverse video conditions.

VI. Results

We tested our pipeline on various videos and scenarios from the CATER dataset, including cases where objects were occluded, contained, or contained and carried. Using this dataset, we evaluated our pipeline's success by measuring prediction accuracy and identifying different types of errors, which are detailed in the quantitative results section. Additionally, we conducted a qualitative analysis to inspect specific errors and explore their potential causes, providing deeper insights into the pipeline's performance and areas for improvement. Our model correctly predicted 24 out of 30 cases in the test set, achieving an **overall accuracy of 80%**. Below is a breakdown of the results by scenario:

Scenario	Correct Predictions	False Predictions	Total Cases	Accuracy (%)
Occluded	11	2	13	84.6
Contained	7	4	11	63.6
Overall	18	6	24	75.0

The model performed better in **occluded cases** (accuracy: **84.6%**) compared to **63.6%** in contained cases, highlighting its strength in handling occlusions while contained scenarios remain challenging due to complex object interactions.

The majority of errors were linked to limitations in the Object Tracking and Object Location Predictor modules. The tracking module struggled with identity consistency across frames, especially during occlusions or complex interactions, due to weaknesses in the object detection model, tracking algorithm, or object embedding. Enhancements in these areas could improve performance. The Object Location Predictor also faced challenges with edge cases and false predictions, suggesting the need to refine its decision trees for greater generalization and accuracy.

Meanwhile, the Video Classification Module performed reliably, meeting expectations and providing a strong foundation. Moving forward, we aim to address weaker components while preserving the strengths of the classification module.

Compared to classic and modified DeepSORT, our pipeline demonstrates notable improvements:

- **Fewer Identity Switches:** Improved consistency in object tracking with fewer color changes.
- **Detection Gaps:** Reduced empty detections by leveraging object permanence predictions to maintain visibility during occlusions or missed detections.

These results affirm the pipeline's enhanced tracking accuracy and robustness in complex scenarios, outperforming prior methods by integrating robust detection, tracking, and object permanence reasoning.

VII. Conclusion

Our pipeline demonstrated promising results in addressing the object permanence task. Unlike other approaches that rely on end-to-end AI models, we presented a modular architecture that offers flexibility, allowing individual components to be modified, replaced, or improved based on specific use cases and scenarios. While the pipeline exhibited some limitations in handling certain complex cases, such as moving occluded objects, the overall architecture performed well on the CATER dataset. This modular design not only provides interpretability but also opens field for future enhancements, making it a adaptable and scalable solution for object permanence and related challenges in computer vision.

VIII. Acknowledgments

We would like to extend our gratitude to the researchers from Bar-Ilan University, Ramat-Gan, Israel, Tel Aviv University, Tel Aviv, Israel, and NVIDIA Research, Tel-Aviv, Israel, for generously sharing their object detection model, which greatly contributed to the development of our pipeline. Their work provided a strong foundation for our research, and we deeply appreciate their support and collaboration.

REFERENCES

[1] Tokmakov, P., Bergmann, P., Meinhardt, T., Leal-Taixé, L., Schiele, B. (2021). Learning to Track with Object Permanence. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 2485-2495. Available: https://arxiv.org/pdf/2103.14258.

[2] Traub, J., Kim, H., Ravi, P., Dube, P., Cho, H. (2024). Loci-Looped: Learning Object Permanence from Videos via Latent Imaginations. Proceedings of the NeurIPS Conference. Available: https://arxiv.org/pdf/2310.10372.

[3] Shamsian, A., Kleinfeld, O., Globerson, A., Chechik, G. (2020). Learning Object Permanence from Video. Proceedings of the European Conference on Computer Vision (ECCV), pp. 1-15. Available: https://arxiv.org/pdf/2003.10469.

[4] Wojke, N., Bewley, A., Paulus, D. (2017). Simple Online and Realtime Tracking with a Deep Association Metric. arXiv preprint. Available: https://arxiv.org/pdf/1703.07402. DOI: https://doi.org/10.48550/arXiv.1703.07402.

[5] Yang, L., Kang, B., Huang, Z., Zhao, Z., Xu, X., Feng, J., Zhao, H. (2024). Depth Anything V2. NeurIPS 2024. Available: https://arxiv.org/pdf/2406.09414. DOI: https://doi.org/10.48550/arXiv.2406.09414.

[6] Girdhar, R., Ramanan, D. (2019). CATER: A Diagnostic Dataset for Compositional Actions and Temporal Reasoning. arXiv preprint. Available: https://arxiv.org/pdf/1910.04744.

[7] Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., Wang, X. (2022). ByteTrack: Multi-Object Tracking by Associating Every Detection Box. arXiv preprint. Available: https://arxiv.org/pdf/2110.06864. DOI: https://doi.org/10.48550/arXiv.2110.06864.

[8] Piaget, J. (1954). The Construction of Reality in the Child. Psychology Press, 1999. 386 pages.

[9] Abouelyazid, M. (2023). Comparative Evaluation of SORT, DeepSORT, and ByteTrack for Multiple Object Tracking in Highway Videos. International Journal of Smart Information and Communication Systems (IJSICS), vol. 8, no. 11, pp. 42–52. Available: https://vectoral.org/index.php/IJSICS/article/view/97/89.

[10] Tong, Z., Song, Y., Wang, J., Wang, L. (2022). VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-Training. NeurIPS 2022 camera-ready version. Available: https://arxiv.org/pdf/2203.12602. DOI: https://doi.org/10.48550/arXiv.2203.12602.

[11] Hugging Face. MCG-NJU/videomae-base Model Documentation. Available: https://huggingface.co/docs/transformers/model$_doc/videomae$.