Tracking through Containers and Occluders in the Wild

Basile Van Hoorick[1] Pavel Tokmakov[2] Simon Stent[3] Jie Li[2] Carl Vondrick[1]
[1]Columbia University [2]Toyota Research Institute [3]Woven Planet
tcow.cs.columbia.edu

Abstract

*Tracking objects with persistence in cluttered and dynamic environments remains a difficult challenge for computer vision systems. In this paper, we introduce **TCOW**, a new benchmark and model for visual tracking through heavy occlusion and containment. We set up a task where the goal is to, given a video sequence, segment both the projected extent of the target object, as well as the surrounding container or occluder whenever one exists. To study this task, we create a mixture of synthetic and annotated real datasets to support both supervised learning and structured evaluation of model performance under various forms of task variation, such as moving or nested containment. We evaluate two recent transformer-based video models and find that while they can be surprisingly capable of tracking targets under certain settings of task variation, there remains a considerable performance gap before we can claim a tracking model to have acquired a true notion of object permanence.*

1. Introduction

The interplay between containment and occlusion can present a challenge to even the most sophisticated visual reasoning systems. Consider the pictorial example in Figure 1a. Given four frames of evidence, where is the red ball in the final frame? Could it be anywhere else? What visual evidence led you to this conclusion?

In this paper, we explore the problem of tracking and segmenting a target object as it becomes occluded or contained by other dynamic objects in a scene. This is an essential skill for a perception system to attain, as objects of interest in the real world routinely get occluded or contained. Acquiring this skill could, for example, help a robot to better track objects around a cluttered kitchen or warehouse [10], or a road agent to understand traffic situations more richly [66]. There are also applications in augmented reality, smart cities, and assistive technology.

It has long been known that this ability, commonly referred to as object permanence, emerges early on in a child's lifetime (see *e.g.* [2, 3, 5–8, 53, 58–61]). But how far away are computer vision systems from attaining the same?

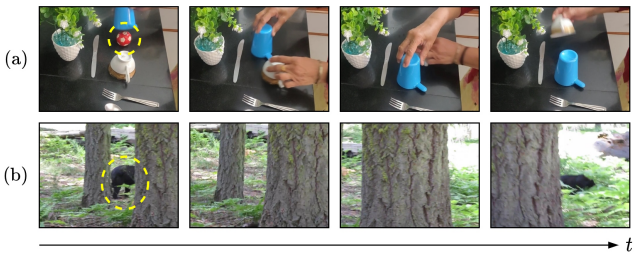


Figure 1. **Containment (a) and occlusion (b)** happen constantly in the real world. We introduce a novel task and dataset for evaluating the object permanence capabilities of neural networks under diverse circumstances.

To support the study of this question, we first propose a comprehensive benchmark video dataset of occlusion- and containment-rich scenes of multi-object interactions. These scenes are sourced from both simulation, in which ground truth masks can be perfectly synthesized, and from the real world, which we hand-annotate with object segments. To allow for an extensive analysis of the behaviours of existing tracking systems, we ensure that our evaluation set covers a wide range of different types of containment and occlusion. For example, even though an object undergoing containment is already a highly non-trivial event, containers can move, be nested, be deformable, become occluded, and much more. Occlusion can introduce considerable uncertainty, especially when the occludee, the occluder, or the camera are in motion on top of everything else.

Using our dataset, we explore the performance of two recent state-of-the-art video transformer architectures, which we repurpose for the task of tracking and segmenting a target object through occlusion and containment in RGB video. We show through careful quantitative and qualitative analyses that while our models achieve reasonable tracking performance in certain settings, there remains significant room for improvement in terms of reasoning about object permanence in complicated, realistic environments. By releasing our dataset and benchmark along with this paper, we hope to draw attention to this challenging milestone on the path toward strong spatial reasoning capabilities.

2. Related Work

Benchmarks for object permanence have begun to appear in our community in recent years, but naturalistic datasets to support this study remain scarce. LA-CATER [57], based on the CATER dataset [29], is a recent example of a synthetic benchmark in which additional localization annotations for the target object were introduced when it is contained, occluded, or carried. More photo-realistic simulation has been applied for studying object permanence in the works of PermaTrack [64], which uses ParallelDomain [1], and 4D dynamic scene completion [66], which relies on CARLA [24].

Notably, most prior datasets and methods focus on localizing occluded objects with a bounding box. In contrast, we focus on a more precise video object segmentation setting. Moreover, rather than attempting to perfectly localize the invisible instance, which is not always possible in practice, we extend the setting of the problem to segmenting the *occluder* instead in ambiguous scenarios. Finally, we introduce a clear distinction between containment and occlusion at the output level.

Object permanence methods in computer vision were mostly studied in the context of multi-object tracking - the task of localizing all the objects from a pre-defined vocabulary with bounding boxes and associating them over time based on identity [28, 45, 46]. As objects only need to be localized when they are visible, occlusions can be handled by simple re-association, but it has been shown that maintaining a hypothesis about the location of invisible objects can help reduce the number of identity switches [13].

To this end, most approaches rely on a simple constant velocity heuristic [15, 47, 75], which propagates the last observed location of an object with a linear motion model. It is, however, only robust when the camera is static and the object velocity does not change significantly during the occlusion (*e.g.* because it is short). More complex, heuristic-based methods include [38, 50], which localize invisible objects by modeling inter-occlusion relationships, and [30] which capitalizes on the correlation between the motion of visible and invisible instances.

More recently, several learning-based methods for localizing invisible objects have been proposed. [57] takes precomputed bounded boxes for visible objects as input and passes them through a recurrent network [35] that is trained to predict the bounding box for the occluded target. In [64] and [63], authors propose end-to-end models that are capable of localizing and associating both visible and invisible instances by capitalizing on a spatiotemporal recurrent memory [9, 39].

Video object segmentation (VOS) is defined as the problem of pixel-accurate separation of foreground objects from the background in videos. In the semi-supervised VOS setting [42, 52], an algorithm is given ground truth masks for objects of interest in the first frame, and has to segment and track them for the rest of the video.

Most existing methods focus on accurately capturing object boundaries and visual appearance rather than modeling complex spatiotemporal phenomena such as object permanence. In particular, the earliest learning-based methods [18, 40, 69] pre-trained a CNN for binary object segmentation on static image datasets, such as COCO [43], and then separately fine-tuned this model on the first frame of the test video for each instance. Evaluating the resulting network on the remaining frames yields fairly strong accuracy, outperforming earlier, heuristic-based methods [4, 26, 32]. However, this approach remains computationally expensive and is not robust to appearance changes, let alone occlusion.

These limitations were later addressed in [19, 37, 72], which replace expensive fine-tuning with cheap matching, and in [44, 51, 68], where online adaptation mechanisms are introduced for modeling the appearance of the target. More recently, memory-based models have become the mainstream approach for video object segmentation [20, 48, 49, 56, 73, 74]. Generally speaking, these methods store feature maps of previous frames together with predicted instance masks in memory. They then retrieve the closest patch with its corresponding label for every patch in the current frame to compute a segmentation.

While these approaches demonstrate impressive performance on existing benchmarks for tracking visible objects, their reliance on visual appearance-based matching means that they cannot segment what they cannot see. In this work, we extend the traditional VOS setting to include segmenting (the occluders and containers of) fully invisible objects, as well as amodally completing partially visible ones [77]. We then evaluate the state-of-the-art AOT approach [73] and demonstrate that it indeed fails in this challenging scenario. Finally, we propose a simple modification of TimeSFormer, a transformer for video [11, 22, 67], to localize both visible and invisible objects, as well as distinguish containment from occlusion.

Sim2real. Leveraging simulated data in machine learning has been essential because real-world data with exhaustive annotation is expensive to scale, or even impossible to acquire. Promising synthetic generators and datasets have been proposed to support various tasks in different domains, including CARLA [24] and ParallelDomain [1] for scene analysis and behavior understanding in autonomous driving, Flying Chairs [23] and Sintel [17] for optical flow, and ThreeDWorld [27] and Kubric [31] for a wide variety of perception tasks in general scenes. We observe a wide variety of data efficiency and sim2real gaps on different tasks. For

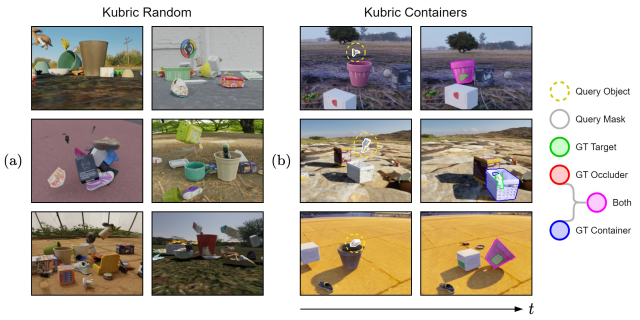


Figure 2. **Simulated datasets.** (a) We show six training examples – these videos consist purely of randomly generated scenes in TCOW Kubric. (b) We show our synthetic benchmark (with annotations) where the actions are scripted – targets fall into containers which are pushed by boxes sliding across the floor and subsequently colliding with them.

example, high generalizability can be observed in low-level feature tasks such as optical flow [62]. On the other hand, for tasks that involve more semantic or global context, synthetic data usually presents a more significant domain gap when transferred to the real world [33, 36, 65]. Thus, selecting the right signal or task to learn from simulation is also critical.

Our experimental results indicate that, without the need for any domain adaptation techniques, reasoning about object persistence by focusing on occluders and containers in simulated environments brings forth a surprisingly promising generalization capacity to the real world, although the overall performance is still below human abilities.

3. Task

In order to tackle object persistence thoughtfully, we propose a methodology that focuses not only on attempting to localize objects at all times, but also prompts models to explicitly consider and decide on possible containers or occluders that might be in the way.

Define $\boldsymbol{x} \in \mathcal{R}^{T\times H\times W\times 3}$ as the RGB-valued input video, and $\boldsymbol{m}_q \in \mathcal{R}^{H\times W}$ as the binary query mask, which perfectly marks the visible pixels belonging to an instance of interest in the first frame. Next, we define the function f, typically a neural network, whose goal is to produce segmentation masks tracking the target object and temporally propagating its mask to densely cover the rest of the video. Unlike traditional VOS settings, though somewhat similarly to [77], f must actually predict a triplet of masks over time:

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{m}_q) = (\hat{\boldsymbol{m}}_t, \hat{\boldsymbol{m}}_o, \hat{\boldsymbol{m}}_c) \tag{1}$$

Here, $\hat{\boldsymbol{m}}_t \in \mathcal{R}^{T\times H\times W}$ is the instance being tracked, $\hat{\boldsymbol{m}}_o \in \mathcal{R}^{T\times H\times W}$ is its frontmost occluder (whenever it exists), and $\hat{\boldsymbol{m}}_c \in \mathcal{R}^{T\times H\times W}$ is its outermost container (whenever it exists). Because the target object always exists *somewhere* (even if located out-of-frame), the ground truth $\boldsymbol{m}_t$ is well-defined for all frames. In contrast, the occluder and container masks, $\boldsymbol{m}_o$ and $\boldsymbol{m}_c$, can be set to all-zero at moments where the target is not occluded or contained.

The triplet of ground truth segmentation masks $(\boldsymbol{m}_t, \boldsymbol{m}_o, \boldsymbol{m}_c)$ ought to fully characterize *all* (*i.e.* visible + invisible) pixels of their respective objects, as if X-ray goggles were provided from the camera's point of view. For this task to become feasible for objects that have become completely hidden, it clearly requires f to learn a notion of object permanence.

However, precisely pinpointing invisible objects is not always possible in practice, which compels us to find a way of dealing with irreducible uncertainty in a principled fashion. Our solution is to ask the model to reveal *which* container or occluder was responsible for enveloping or hiding a target object, leading to a more expressive and interpretable representation. Figure 2b illustrates this concept.

Finally, in order to operate in a causal (online) fashion, the predicted set of masks $\hat{\boldsymbol{y}}_t$ at any time $t \in [1, T]$ may depend only on all past input frames $\boldsymbol{x}_{\leq t}$ up until the present.

3.1. Evaluation Metrics

We report the mean IoU (Intersection over Union) score, also known as region-based segmentation similarity or Jaccard index $\mathcal{J}$ [52]. In VOS, this is a conventional measure of how well a confidence-thresholded prediction overlaps with the ground truth mask [20, 73], and thus how well the model succeeds at accurately tracking the queried object of interest throughout the video.

For a sequence of target object masks $\hat{\boldsymbol{m}}_t$, the resulting IoU $\mathcal{J}_{target}$ is averaged over all frames. For the occluder and container masks $\hat{\boldsymbol{m}}_o$ and $\hat{\boldsymbol{m}}_c$, the respective IoU values $\mathcal{J}_{occl}$ and $\mathcal{J}_{cont}$ are averaged only over those frames where an occluder or container actually exists in the video. In terms of ground truth annotations, formal definitions as to how we determine occlusion and containment events in our framework are given in Section 5.

When evaluating multiple clips, whereas $\mathcal{J}_{target}$ is averaged uniformly across scenes, both $\mathcal{J}_{occl}$ and $\mathcal{J}_{cont}$ are weighted-averaged according to how many samples were measured per video for each type. This ensures that challenging examples with more or longer-term occlusions will be weighted more heavily than less cluttered videos where none or only a handful of frames have an active occluder.

4. Datasets

To bring our proposed task to life, we introduce a new collection of datasets with the intent to facilitate both learning and evaluating object permanence. Our data is derived from synthetic sources (**TCOW Kubric**) as well as the real world (**TCOW Rubric**). While Kubric manifests dense,

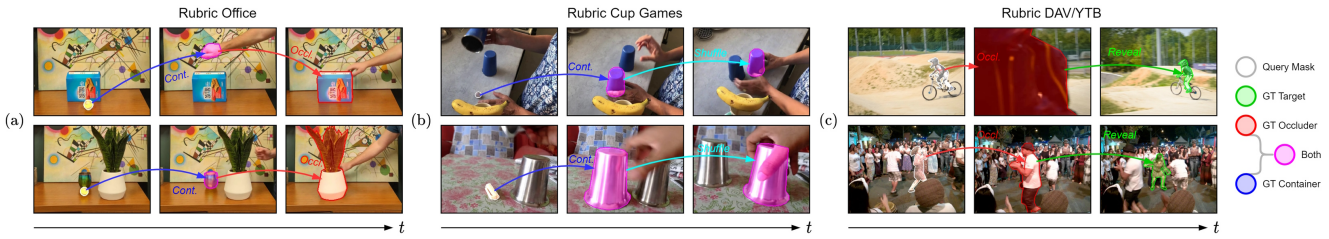


Figure 3. **Real-world benchmark.** We show six examples (with ground truth annotations) from Rubric Office (a), Rubric Cup Games (*i.e.* DeepMind Perception Test [55]) (b), and Rubric DAV/YTB (*i.e.* DAVIS [54] and YouTube-VOS [70]) (c). Here, a white outline denotes the query mask in the first frame. A red outline denotes the main occluder in front of a (different) target object, and a blue outline denotes the main container surrounding a target object, such that a magenta outline implies that one and the same object is responsible for both occluding and containing the target. Finally, a green outline denotes the target instance itself when it re-emerges.

TCOW Dataset	S/R	# Videos	# Frames / vid.	Resolution	# Masks / vid.	# Cont. events / vid.	# Occl. events / vid.
Kubric Random	Sim	4000	36	480 × 360	180-1188	0–1	0–4
Kubric Containers	Sim	27	36	480 × 360	180-252	1	0–2
Rubric Office	Real	32	150–330	640 × 480	3-6	0–2	0–4
Rubric Cup Games	Real	14	308–463	640 × 480	4-14	1–3	0–3
Rubric DAV/YTB	Real	33	41–180	640 × 480	2-9	0	0–5

Table 1. **Dataset properties.** TCOW consists of five parts. Kubric Random has a train/val/test split of 3600/200/200 scenes, and all other datasets are strictly test sets for the purpose of evaluation. The number of containment or occlusion events is incremented every time a potential target object enters a container or goes behind an occluder respectively.

exact annotations useful for training, Rubric comprises a novel challenging benchmark for understanding object permanence in the wild. Relevant statistics are summarized in Table 1.

4.1. Kubric

We leverage the Kubric [31] simulator as the synthetic data generator for all training data, plus some evaluation videos. We modify the provided *MOVi-F* template to insert containers more often, which are sourced randomly from a manually predefined list of assets within Google Scanned Objects [25]. Every scene has between 6 and 36 objects in total; roughly one-third of them are spawned in mid-air at the beginning of the video.

To construct the X-ray segmentation mask $\boldsymbol{m}_a \in [0,1]^{T\times H\times W\times K}$, we collect raw ground truth masks over time for all pixels of all K instances separately. In addition, we study all *pairs* of objects to derive any possible container-containee or occluder-occludee relationships that might emerge. Because we have access to perfect information in a simulated environment, the annotation framework described in Section 5 can be applied directly.

As shown in Figure 2, we procedurally two generate versions of the TCOW Kubric dataset. First, *Kubric Random* consists of a large number of cluttered scenes where the objects are spawned with independent, random velocities, thus causing various collisions and complex interactions to emerge. Occlusion and containment frequently happen by chance, encouraging neural networks to learn spatial reasoning skills and motion patterns from data.

Second, *Kubric Containers* is a more constrained set of scripted videos, each of which portrays a single object falling into a container that subsequently gets pushed and displaced by a third object, *i.e.* a moving box, that had been spawned simultaneously with a high initial horizontal velocity. Because annotations are cheap in simulation, this is the most densely labeled evaluation set.

4.2. Rubric

To support effective real-world evaluations, we introduce TCOW Rubric, a diverse collection of naturalistic videos depicting open-world objects experiencing containment and occlusion in various circumstances, with distinct levels of difficulty. Our data is sourced internally from videos recorded in an office space (*Rubric Office*), as well as externally from DeepMind Perception Test [55] (*Rubric Cup Games*), DAVIS 2017 [54], and YouTube-VOS 2019 [70] (*Rubric DAV/YTB*). Figure 3 showcases a few examples of our three real-world datasets.

5. Labeling for Object Permanence

For evaluation and training purposes, we wish to define and distinguish occlusion and containment events when they occur. In practice however, the state of whether an object is being occluded or being contained by another is not always clear-cut, because both concepts can be treated as a spectrum. In the following discussion, a so-called *occluder-*

occludee or *container-containee* relationship refers to a putative object (that is not the target itself) acting as an occluder or container, *i.e.* it is responsible for either hiding or encompassing the target instance of interest, denoted the occludee or containee respectively. A clear formalism is required, which we first describe in the context of simulated data, where perfect information is available.

Unlike occlusion, we regard containment as being fundamentally a 3D phenomenon, because the fact that one object is inside another can generally be stated independently of camera viewpoints. In contrast, occlusions are by definition *purely* a function of perspective projections to 2D images. Hence, occlusion and containment exist as separate principles and are also calculated in different ways.

5.1. Visible versus X-ray annotations

Consider a dynamic scene with K (not necessarily unique) objects, such that any recorded video $\boldsymbol{x} \in \mathcal{R}^{T\times H\times W\times 3}$ will visually depict up to K objects plus the background. Define $\boldsymbol{m}_v \in [0, K]^{T\times H\times W}$ as an integer-valued *visible segmentation mask* over time that marks the 1-based instance ID for each pixel in $\boldsymbol{x}$, where 0 is reserved for the background. Define $\boldsymbol{m}_a \in [0, 1]^{T\times H\times W\times K}$ as a binary-valued *X-ray segmentation mask* over time. That is, per frame $t \in [1, T]$ and per object index $k \in [1, K]$, the pixels in $\boldsymbol{m}_a$ are essentially boolean indicators of whether hypothetical rays emanating from the camera would hit instance k at least once if it were the only object in existence.[1] An arbitrary combination of objects can reside along a single ray, implying that in principle, any binary pattern is possible along the last dimension of $\boldsymbol{m}_a$. In particular, all values will be zero if and only if that pixel is part of the background.

5.2. Quantifying Occlusion

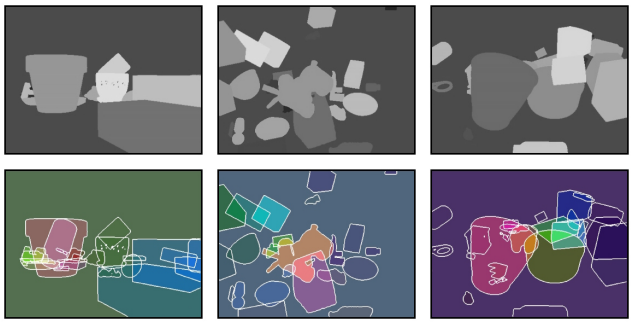
Assuming the occludee has a well-defined boundary mask, there exist varying levels of occlusion by any occluder, and we approximate this by measuring and comparing the number of *visible* versus *total* (*i.e.* visible + invisible) pixels. Specifically, the *occlusion fraction* (or percentage) $o_{k,t} \in [0, 1]$ for instance k at time t is defined as follows:

$$o_{k,t} = 1 - \frac{\sum_{x,y} \mathbb{1}\left[\boldsymbol{m}_v(t, y, x) = k\right]}{\sum_{x,y} \boldsymbol{m}_a(t, y, x, k)} \tag{2}$$

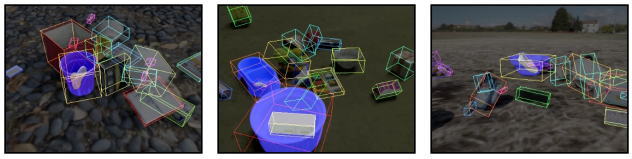
where $\boldsymbol{m}_v \in [0, K]^{T\times H\times W}$ and $\boldsymbol{m}_a \in [0, 1]^{T\times H\times W\times K}$ are the previously defined visible and X-ray segmentation masks respectively, illustrated in Figure 4a.

We choose a threshold of 95%, which means that whenever the occlusion fraction satisfies $o_{k,t} \geq 0.95$, then k is said to be *invisible*. Moreover, whichever object l has the most (visible) pixels in front of instance k is designated as its *main occluder* at time t, and consequently populates the ground truth occluder mask $\boldsymbol{m}_o$ for that frame.

[1] Note that every object in isolation is treated purely as the sum of its pixels from a 2D perspective. As such, *intra*-object phenomena such as self-occlusion are ignored in this paper in favor of *inter*-object phenomena.



(a) **Detecting occlusion** takes place by comparing visible segmentation masks $\boldsymbol{m}_v$ (top) with X-ray segmentation masks $\boldsymbol{m}_a$ (bottom – colors are assigned randomly) to (1) identify which objects have become invisible at any point in time, and (2) for every such event, find out exactly which occluder is responsible.



(b) **Detecting containment** occurs by comparing the 3D bounding boxes between all pairs of instances, revealing when a smaller object (marked in gray) is located inside a concave larger object (marked in blue).

Figure 4. Visualizations for understanding the methodology for gathering ground truth information with respect to inter-object interactions that pertain to object persistence.

5.3. Quantifying Containment

We define $b_{k,t} \in \mathcal{R}^{3\times 8}$ as the spatial world coordinates of the eight corners of the 3D bounding box of instance k at time t, illustrated in Figure 4b.[2] For every other object l, the pair-wise *containment fraction* (or percentage) $c_{k,l,t}$ between a containee k and its putative container l is defined as follows:

$$c_{k,l,t} = \frac{|b_{k,t} \cap b_{l,t}|}{|b_{k,t}|} \tag{3}$$

where $\cap$ is the geometric intersection operator, and $|b_{k,t}|$ denotes the physical 3D volume of the cuboid enveloping instance k.[3]

[2] These coordinates are embedded in a shared frame of reference with respect to the center of the Kubric scene, although the boxes themselves are not axis-aligned – instead, they follow the canonical object frame and rotate along with its pose.

[3] For example, $|b_{k,t}|$ can be calculated as the absolute value of the determinant of the matrix containing the three basis vectors spanning the 3D cuboid associated with $b_{k,t}$. As for $b_{k,t} \cap b_{l,t}$ however, it is non-trivial in practice to measure volumes of the arbitrary polyhedra that may arise from intersecting two unaligned cuboids, so we instead approximate this value by densely sampling points inside $b_{k,t}$ and calculating the fraction of them that also reside within $b_{l,t}$.

We choose a threshold of 75%, which means that whenever $c_{k,l,t} \geq 0.75$ (*i.e.* more than 75% of the target's volume is enclosed by a container l), then l is designated as the *main container* of k, populating the ground truth container mask $\boldsymbol{m}_c$ for that frame.

Rarely, we need to disambiguate multiple candidate containers $\{l_1, \ldots, l_n\}$, with $c_{k,l_i,t} \geq 0.75, \forall i \in [1, n]$. This can happen *e.g.* in the case of nested containment if k is the innermost object. In this case, we search for whichever l_i is the "least contained" by any other l_j, and is as such the *outermost* container surrounding k as well as all other candidates. Specifically, the main container l_i of the target instance k at time t is defined by the solution to the optimization problem $i = \min_i \max_j c_{l_i,l_j,t}$.

5.4. Annotations in the real world

While it is possible (albeit expensive) to obtain visible segmentation masks $\boldsymbol{m}_v$ in natural videos via human annotation, accurate X-ray segmentation masks $\boldsymbol{m}_a$ for cluttered scenarios can typically only feasibly be retrieved via simulation due to inherent ambiguity.

For our Rubric datasets, we first select a sparse subset of key frames depicting salient moments of interest in each video. Then, we manually label these moments with either a target mask $\boldsymbol{m}_t$ when the object is fully or mostly visible, or a frontmost occluder when it is nearly completely invisible, and/or an outermost container mask when it is fully enclosed by (the convex hull of) another object. All annotations, except where DAVIS or YouTube-VOS provided them already, were drawn by a single expert annotator by filling roughly a dozen connected line segments.

6. Experiments

In this section, we evaluate two state-of-the-art, transformer-based neural network models, in addition to several heuristics that use ground truth annotations to generate predictions. We report how well each baseline performs on both synthetic and real-world data, and analyze the main trends in success versus failure cases.

6.1. Baseline Models

AOT: Video object segmentation (VOS) is perhaps the most similar task to our own, so we adopt the competitive Associating Objects with Transformers (AOT) method [73] as-is and retrain it on Kubric to teach it to track through occlusions, *i.e.* to produce the target mask $\hat{\boldsymbol{m}}_t$ from an input video x and query mask $\boldsymbol{m}_q$. We take an AOT-B checkpoint pretrained on static images (see [73] for details), and retrain the network on the training split of Kubric Random. However, since AOT is originally trained on YouTube-VOS and, therefore, already capable of segmenting objects in video, we also evaluate a plug-and-play variant of AOT-B without any further learning.

TCOW (Ours): For our second baseline, we customize the competitive TimeSFormer model [11] as backbone to predict a triplet of masks $(\hat{\boldsymbol{m}}_t, \hat{\boldsymbol{m}}_o, \hat{\boldsymbol{m}}_c)$ given $(\boldsymbol{x}, \boldsymbol{m}_q)$, instead of a category. We leverage its attention-based spatiotemporal context modeling capabilities and treat the output sequence as a feature map for dense video segmentation. Specifically, we ignore the classification token in favor of a linear projection from the set of embeddings after the last self-attention block back to a set of image patches of size $16 \times 16 \times 3$ that, when spatially recombined together, constitute the predictions for target, occluder, and container masks. To ensure a fair comparison with AOT, we apply a causal mask to the attention weights inside the temporal self-attention block, to prevent information from leaking backward in time during inference. Following [11], we initialize the network weights with a ViT-Base [22] ImageNet-pretrained checkpoint, and similarly retrain it on the training split of Kubric Random.

6.2. Baseline Heuristics

Video instance segmentation (VIS) [71] is another closely related task. We introduce oracle baselines that have access to perfect visible instance segmentation masks and track target objects or their occluders or containers by selecting the appropriate instance from the ground truth annotations. While varying levels of thoroughness exist in imitating and repurposing expert VIS models toward object permanence, we choose the following four in order of increasing complexity:

Copy query: Since VOS models can see the ground truth label associated with the first frame, a simple baseline is to propagate this mask to future frames without changing it.

Static mask (during occlusion): The target object is segmented perfectly whenever visible or partially occluded. During full occlusions (as defined in Section 5), we copy and propagate the last non-occluded ground truth X-ray mask, and hold it in that location until it re-emerges again, at which point we continue the perfect tracking routine.

Linear extrapolation (during occlusion): This baseline is an extension of *Static mask* that explicitly encodes and implements the constant velocity assumption that is often used as a prior in earlier works [15, 41, 47, 64, 75]. When the target instance enters a total occlusion at time t, its center of gravity in the two preceding frames is used to estimate an instantaneous speed vector, which is used to propagate the ground truth X-ray mask from frame t until the next disocclusion occurs, at which point we return to perfect tracking.

Method	Training set	Kubric Random (test set)				Kubric Containers			
		$\mathcal{J}_{tgt,all}$	$\mathcal{J}_{tgt,invis}$	$\mathcal{J}_{occl}$	$\mathcal{J}_{cont}$	$\mathcal{J}_{tgt,all}$	$\mathcal{J}_{tgt,invis}$	$\mathcal{J}_{occl}$	$\mathcal{J}_{cont}$
AOT (direct plug)	Static + YouTube-VOS	30.4	0.4	0.5*	1.3*	22.5	0.9	4.6*	2.3*
AOT (visible only)	Static + Kubric	35.0	0.5	0.7*	1.4*	23.1	0.7	2.0*	1.9*
AOT (cartoon)	Static + Kubric (flat)	29.8	5.4	3.7*	4.1*	20.4	0.9	4.2*	4.7*
AOT [73]	Static + Kubric	41.3	6.8	5.1*	4.9*	26.5	2.5	6.8*	5.9*
TCOW (visible only)	ImageNet + Kubric	44.7	0.1	64.6	60.0	25.2	0.1	73.9	76.3
TCOW (cartoon)	ImageNet + Kubric (flat)	31.3	5.6	30.0	43.6	21.7	2.3	26.2	40.1
TCOW	ImageNet + Kubric	**53.0**	**16.6**	**70.5**	**71.6**	**36.8**	**16.0**	**76.8**	**78.2**
Copy query	-	5.8	0.4	-	-	7.8	0.5	-	-
Static mask†	-	58.3†	10.1†	-	-	39.3†	10.2†	-	-
Linear extrapolation†	-	59.8†	15.6†	-	-	39.6†	10.8†	-	-
Jump to occluder†	-	48.1†	-	69.3†	-	32.5†	-	87.2†	-

Table 2. **Results in TCOW Kubric (synthetic).** We report the average IOU [%] per frame (higher is better). Our TCOW model outperforms most other baselines and ablations, and can mark both containers and occluders even more accurately than the target object itself. *Since AOT is incapable of outputting multiple masks for a single query instance, we compare the same prediction with all three ground truths. †Heuristic that uses privileged information, *i.e.* can access ground truth annotations.

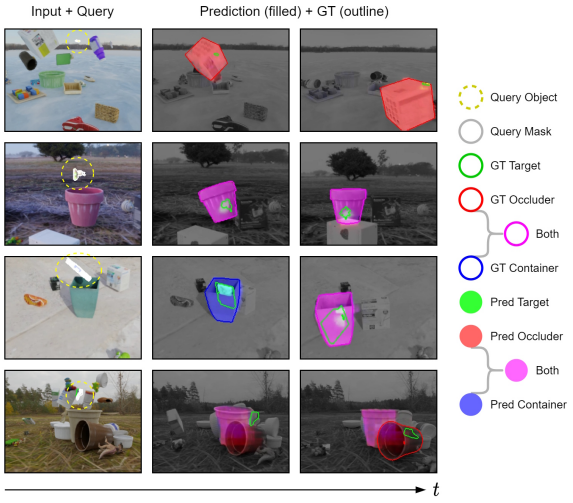


Figure 5. **Qualitative results for TCOW Kubric (synthetic).** All visualized predictions are made by the TCOW network. The first column shows the first frame along with the query mask highlighting the object we wish to track. The query object can be tiny, so we encircle it with a yellow dashed line for clarity.

Jump to occluder: The target object is segmented perfectly to produce $\hat{\boldsymbol{m}}_t$ until the first full occlusion occurs at time t. Then, whichever instance has the highest number of visible pixels in front of the target's ground truth X-ray mask (*i.e.* its main occluder) takes on the role of the object to track starting at frame t, filling in $\hat{\boldsymbol{m}}_o$.[4] This heuristic is similar in nature to switching to tracking the nearest object when the current one has been lost, but is more powerful as it assumes knowledge of the responsible ground truth occluder. It is the most advanced heuristic in the sense that it explicitly considers and populates the occluder mask $\boldsymbol{m}_o$, while the previous three heuristics pertain to the target mask $\boldsymbol{m}_t$ only.

[4]Since this occluder could potentially itself also become occluded by yet another object, the described procedure may be applied recursively.

6.3. Model Ablations

Foreshadowing decent results, it is worth asking where a model's performance and generalization ability comes from in the context of object permanence.

Visible pixels only: *How important is the ability to access and directly use X-ray annotations as ground truth masks for learning to track with object permanence?* We study how the results change if we supervise models with only the visible parts of target objects and putative occluders or containers.

Cartoon training data: *How important is it to ensure a faithful visual appearance of scenes when learning to track with object permanence?* Visual realism, or the lack thereof, is often a cause for concern when working with synthetic data. While no perfect simulator exists, Kubric boasts a respectable degree of realism. Hence, we wish to examine how influential this aspect really is. To make Kubric look significantly less photorealistic, we turn off all textures by uniformly replacing all objects with unique, randomly chosen colors, as if every frame was replaced with its visible instance segmentation mask $\boldsymbol{m}_v$.

6.4. Results

Table 2 shows quantitative results on simulated data. $\mathcal{J}_{tgt,all}$ represents the mean Jaccard index of the target instance over all frames, but to study the localization performance of hidden objects, $\mathcal{J}_{tgt,invis}$ considers only frames where the target is fully occluded by another object. On average, both AOT and TCOW perform somewhat similarly in terms of segmenting the target object, although TCOW shines in recognizing the correct occluder or container whenever the target becomes occluded or contained

Method	Training set	Rubric Office			Rubric Cup Games			Rubric DAV/YTB	
		$\mathcal{J}_{target}$	$\mathcal{J}_{occl}$	$\mathcal{J}_{cont}$	$\mathcal{J}_{target}$	$\mathcal{J}_{occl}$	$\mathcal{J}_{cont}$	$\mathcal{J}_{target}$	$\mathcal{J}_{occl}$
AOT (direct plug)	Static + YouTube-VOS	**78.2**	5.3*	8.2*	41.7	3.3*	4.3*	**63.4**	8.6*
AOT (visible only)	Static + Kubric	58.0	4.8*	6.9*	44.7	4.5*	4.6*	51.9	10.0*
AOT (cartoon)	Static + Kubric (flat)	45.6	2.9*	3.0*	38.6	10.6*	9.6*	44.5	11.2*
AOT [73]	Static + Kubric	54.1	6.4*	8.0*	**50.2**	13.1*	11.8*	50.8	12.7*
TCOW (visible only)	ImageNet + Kubric	72.5	**39.2**	**12.5**	34.8	27.6	3.5	51.3	31.6
TCOW (cartoon)	ImageNet + Kubric (flat)	35.7	12.1	7.7	31.9	8.8	**14.3**	22.4	9.2
TCOW	ImageNet + Kubric	69.4	30.1	11.7	38.3	**35.0**	7.6	52.8	**33.4**
Copy query	-	12.5	-	-	18.6	-	-	15.8	-

Table 3. **Results in TCOW Rubric (real-world).** We report the average IOU [%] per frame (higher is better). *Since AOT is incapable of predicting multiple masks for a single query instance, we compare the same output with all three ground truths.

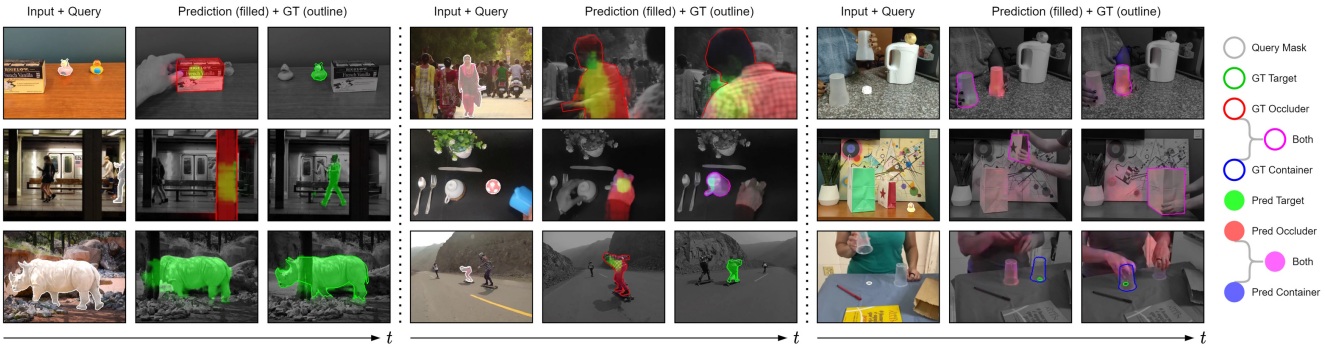


Figure 6. **Qualitative results for TCOW Rubric (real-world).** All visualized predictions are made by the TCOW network. We show six success cases in the left and middle columns, and three failure cases on the right.

respectively. The most privileged baseline algorithm (*Jump to occluder*) also works well for many cases, but is incapable of distinguishing containment from occlusion.

Figure 5 demonstrates several examples produced by TCOW, the best-performing model, on Kubric data. In most cases, the main containers or occluders responsible for surrounding or concealing the target are segmented very accurately in nearly all frames.

Table 3 shows real-world numerical results, categorized by data source.[5] Because AOT is the result of years of optimization by the VOS community, it boasts strong results for segmenting target objects, especially visible ones. However, it is trained with only a relatively short context of 5 frames, which works well for conventional VOS, but seems to break down in terms of longer-term spatiotemporal reasoning, which is required for object permanence.

The decent performance of the 'TCOW (visible only)' ablation suggests that it is often more fruitful to track the surrounding occluder or container of a fully hidden target object k rather than to try to precisely localize k at all times, which supports our task definition in Section 3. Moreover, the fair performance of the 'TCOW (cartoon)' ablation suggests that learning the correct motion signals and occlusion/containment dynamics is important for capturing object permanence, and the remaining gap is filled by adding more realism.

Figure 6 shows representative success cases and failure cases made by the non-ablated TCOW model on real-world data. In general, this network performs surprisingly well – for example, total occlusions involving occludees and/or occluders far outside of the training distribution are often still handled fairly correctly. For partially occluded instances, such as the rhino on the lower left, a solid amodal completion capability is demonstrated as well.

However, there exist many Rubric videos where both models break down almost completely. Comparing Table 2 with Table 3, the quality of the occluder and container masks drop substantially when moving from synthetic to real data. Containment in particular appears to be the more difficult concept to learn robustly [34]. We qualitatively observe that recursive containment, as exemplified with paper bags going inside one another in Figure 6 (center right), is among the toughest to tackle. In fact, there is not a single such example in Rubric that is addressed satisfactorily. Tracking objects through containment by upside-down cups that are repeatedly shuffled around also turns out to be highly demanding, especially when the cups are identical. Lastly, videos with transparent containers present yet another failure scenario, presumably because non-opaque objects do not exist in the Kubric training data.

[5] There is no $\mathcal{J}_{tgt,invis}$ metric because fully occluded objects are never labeled in the real world; only their occluders are.

7. Discussion

In this work, we propose the challenging TCOW benchmark, which in its totality covers many different types of containment and occlusion, including compositions thereof. The TCOW model, based on TimeSFormer, shows promising yet lacking performance, and we believe future tracking models ought to address and resolve these scenarios more effectively. While we have made significant strides in solving elementary base cases of occlusion and containment, object permanence as a whole remains far from being solved. We, therefore, invite and encourage the community to work on this problem.

Acknowledgements: We thank Revant Teotia, Ruoshi Liu, Scott Geng, and Sruthi Sudhakar for helping record TCOW Rubric videos. This research is based on work partially supported by the Toyota Research Institute, the NSF CAREER Award #2046910, and the NSF Center for Smart Streetscapes (CS3) under NSF Cooperative Agreement No. EEC-2133516. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

[1] Parallel domain: Data pipeline for computer vision. https://paralleldomain.com/, March 2021. 2

[2] Andréa Aguiar and Renée Baillargeon. 2.5-month-old infants' reasoning about when objects should and should not be occluded. *Cognitive psychology*, 39(2):116–157, 1999. 1

[3] Andréa Aguiar and Renée Baillargeon. Developments in young infants' reasoning about occluded objects. *Cognitive psychology*, 45(2):267–336, 2002. 1

[4] S Avinash Ramakanth and R Venkatesh Babu. Seamseg: Video object segmentation using patch seams. In *CVPR*, 2014. 2

[5] Renee Baillargeon. Representing the existence and the location of hidden objects: Object permanence in 6-and 8-month-old infants. *Cognition*, 23(1):21–41, 1986. 1

[6] Renee Baillargeon. Object permanence in $3 1/2$-and $4 1/2$-month-old infants. *Developmental psychology*, 23(5):655, 1987. 1

[7] Renée Baillargeon and Julie DeVos. Object permanence in young infants: Further evidence. *Child development*, 62(6):1227–1246, 1991. 1

[8] Renee Baillargeon, Elizabeth S Spelke, and Stanley Wasserman. Object permanence in five-month-old infants. *Cognition*, 20(3):191–208, 1985. 1

[9] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. In *ICLR*, 2016. 2

[10] Wissam Bejjani, Wisdom C Agboh, Mehmet R Dogar, and Matteo Leonetti. Occlusion-aware search for object retrieval in clutter. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4678–4685. IEEE, 2021. 1

[11] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, 2021. 2, 6, 12, 13

[12] Jeroen Bertels, Tom Eelbode, Maxim Berman, Dirk Vandermeulen, Frederik Maes, Raf Bisschops, and Matthew B Blaschko. Optimizing the dice score and jaccard index for medical image segmentation: Theory and practice. In *International conference on medical image computing and computer-assisted intervention*, pages 92–100. Springer, 2019. 13

[13] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 2

[14] Blender Online Community. Blender - a 3d modelling and rendering package, 2021. 12

[15] Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *ICCV*, 2009. 2, 6

[16] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 12, 13

[17] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *European conference on computer vision*, pages 611–625. Springer, 2012. 2

[18] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *CVPR*, 2017. 2

[19] Yuhua Chen, Jordi Pont-Tuset, Alberto Montes, and Luc Van Gool. Blazingly fast video object segmentation with pixel-wise metric learning. In *CVPR*, 2018. 2

[20] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022. 2, 3

[21] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. 2016. 12

[22] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 6, 12

[23] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[24] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. 2

[25] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh,

and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. *arXiv preprint arXiv:2204.11918*, 2022. 4, 12

[26] Qingnan Fan, Fan Zhong, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Jumpcut: non-successive mask transfer and interpolation for video cutout. *ACM Trans. Graph.*, 34(6):195–1, 2015. 2

[27] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020. 2

[28] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012. 2

[29] Rohit Girdhar and Deva Ramanan. CATER: A diagnostic dataset for compositional actions and temporal reasoning. In *ICLR*, 2020. 2

[30] Helmut Grabner, Jiri Matas, Luc Van Gool, and Philippe Cattin. Tracking the invisible: Learning where the object might be. In *CVPR*, 2010. 2

[31] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3749–3761, 2022. 2, 4, 12

[32] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*. IEEE, 2010. 2

[33] Vitor Guizilini, Jie Li, Rareș Ambruș, and Adrien Gaidon. Geometric unsupervised domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8537–8547, 2021. 3

[34] Susan J Hespos and Renée Baillargeon. Infants' knowledge about occlusion and containment events: A surprising discrepancy. *Psychological Science*, 12(2):141–147, 2001. 8

[35] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[36] Han-Kai Hsu, Chun-Han Yao, Yi-Hsuan Tsai, Wei-Chih Hung, Hung-Yu Tseng, Maneesh Singh, and Ming-Hsuan Yang. Progressive domain adaptation for object detection. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 749–757, 2020. 3

[37] Yuan-Ting Hu, Jia-Bin Huang, and Alexander G Schwing. Videomatch: Matching based video object segmentation. In *ECCV*, 2018. 2

[38] Yan Huang and Irfan Essa. Tracking multiple objects through occlusions. In *CVPR*, 2005. 2

[39] Allan Jabri, Andrew Owens, and Alexei A Efros. Space-time correspondence as a contrastive random walk. In *NeurIPS*, 2020. 2

[40] Anna Khoreva, Rodrigo Benenson, Eddy Ilg, Thomas Brox, and Bernt Schiele. Lucid data dreaming for video object segmentation. *International Journal of Computer Vision*, 127(9):1175–1197, 2019. 2

[41] Tarasha Khurana, Achal Dave, and Deva Ramanan. Detecting invisible people. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3174–3184, 2021. 6

[42] Fuxin Li, Taeyoung Kim, Ahmad Humayun, David Tsai, and James M Rehg. Video segmentation by tracking many figure-ground segments. In *ICCV*, 2013. 2

[43] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 2

[44] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. In *ACCV*, 2018. 2

[45] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. Multiple object tracking: A literature review. *Artificial Intelligence*, page 103448, 2020. 2

[46] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 2

[47] Dennis Mitzel, Esther Horbert, Andreas Ess, and Bastian Leibe. Multi-person tracking with sparse detection and continuous segmentation. In *ECCV*, 2010. 2, 6

[48] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *CVPR*, 2018. 2

[49] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 2

[50] Vasilis Papadourakis and Antonis Argyros. Multiple objects tracking in the presence of long-term occlusions. *Computer Vision and Image Understanding*, 114(7):835–846, 2010. 2

[51] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images. In *CVPR*, 2017. 2

[52] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 724–732, 2016. 2, 3

[53] Jean Piaget. *The construction of reality in the child*. Routledge, 2013. 1

[54] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 4

[55] Viorica Pătrăucean, Lucas Smaira, Ankush Gupta, Adrià Recasens Continente, Larisa Markeeva, Dylan Banarse, Mateusz Malinowski, Yi Yang, Carl Doersch, Tatiana Matejovicova, Yury Sulsky, AntoineMiech, Skanda Koppula, Alex Frechette, Hanna Klimczak, Raphael Koster, Junlin Zhang, StephanieWinkler, Yusuf Aytar, Simon Osindero, Dima Damen, Andrew Zisserman, and João Carreira. Perception Test: A Diagnostic Benchmark for Multimodal Models. Technical report, DeepMind, 10 2022. 4

[56] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Kernelized memory network for video object segmentation. In *ECCV*, 2020. 2

[57] Aviv Shamsian, Ofri Kleinfeld, Amir Globerson, and Gal Chechik. Learning object permanence from video. In *ECCV*, 2020. 2

[58] Elizabeth Spelke. Initial knowledge: Six suggestions. *Cognition*, 50(1-3):431–445, 1994. 1

[59] Elizabeth S Spelke. Principles of object perception. *Cognitive science*, 14(1):29–56, 1990. 1

[60] Elizabeth S Spelke. Where perceiving ends and thinking begins: The apprehension of objects in infancy. In *Perceptual development in infancy*, pages 209–246. Psychology Press, 2013. 1

[61] Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007. 1

[62] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 3

[63] Pavel Tokmakov, Allan Jabri, Jie Li, , and Adrien Gaidon. Object permanence emerges in a random walk along memory. In *ICML*, 2022. 2

[64] Pavel Tokmakov, Jie Li, Wolfram Burgard, and Adrien Gaidon. Learning to track with object permanence. In *ICCV*, 2021. 2, 6

[65] Marco Toldo, Andrea Maracani, Umberto Michieli, and Pietro Zanuttigh. Unsupervised domain adaptation in semantic segmentation: a review. *Technologies*, 8(2):35, 2020. 3

[66] Basile Van Hoorick, Purva Tendulkar, Didac Suris, Dennis Park, Simon Stent, and Carl Vondrick. Revealing occlusions with 4d neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3011–3021, 2022. 1, 2

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 12

[68] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *BMVC*, 2017. 2

[69] Huaxin Xiao, Jiashi Feng, Guosheng Lin, Yu Liu, and Maojun Zhang. Monet: Deep motion exploitation for video object segmentation. In *CVPR*, 2018. 2

[70] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *ECCV*, 2018. 4

[71] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5188–5197, 2019. 6

[72] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K. Katsaggelos. Efficient video object segmentation via network modulation. *CVPR*, 2018. 2

[73] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating objects with transformers for video object segmentation. *Advances in Neural Information Processing Systems*, 34:2491–2502, 2021. 2, 3, 6, 7, 8, 12, 13

[74] Zongxin Yang, Yunchao Wei, and Yi Yang. Collaborative video object segmentation by multi-scale foreground-background integration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[75] Qian Yu, Gérard Medioni, and Isaac Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In *CVPR*, 2007. 2, 6

[76] Greg Zaal, Rob Tuytel, Rico Cilliers, James Ray Cock, Andreas Mischok, Sergej Majboroda, Dimitrios Savva, and Jurita Burger. Polyhaven: a curated public asset library for visual effects artists and game designers. https://polyhaven.com/hdris, 2021. 12

[77] Guanqi Zhan, Weidi Xie, and Andrew Zisserman. A tri-layer plugin to improve occluded detection. *arXiv preprint arXiv:2210.10046*, 2022. 2, 3

Tracking through Containers and Occluders in the Wild

Supplementary Material

A. Dataset Details

For our training set TCOW Kubric Random, all scenes are generated based on the *MOVi-F*[6] template code [31], but with several modifications. Backgrounds are chosen randomly from the Polyhaven HDRI collection [76], and all objects originate from Google Scanned Objects (GSO) [25]. Every scene spawns s static objects lying on the ground, and d dynamic objects falling down when the video starts. s is uniformly randomly chosen between 4 and 24 (inclusive), while d is uniformly randomly chosen between 2 and 12 (inclusive).

In order to increase the frequency of containment, we manually scan the GSO library to designate 114 out of 1,032 GSO assets as *containers*, which can be either deep or shallow. For each scene, at least three out of the s static objects must be containers, and while object sizes are chosen randomly, we also make containers slightly bigger on average. An assortment of examples is shown in Figure 7.

The most time-consuming part of the simulation is generating the X-ray segmentation mask $\boldsymbol{m}_a \in [0,1]^{T\times H\times W\times K}$, which is used for supervision as it exposes all pixels of all K instances separately over time, regardless of occlusion. This is done by running the PyBullet physics simulation [21] once, thus letting the object interactions develop over time within the dynamic scene, then rendering the input video via Blender [14] with all instances present, following [31]. Next, we isolate each object by turning off the visibility of all other objects (they are essentially temporarily removed from existence), and rendering those videos again separately to iteratively produce one channel of $\boldsymbol{m}_a$ at a time.

Finally, even though the frame rate of video clips in the Rubric benchmark is variable (*i.e.* between 4 and 30), rendering of all Kubric simulations happens at a single fixed value of 12 FPS.

To construct the Kubric Random dataset, consisting of 4,000 videos of 36 frames each with spatial dimension 480×360 along with RGB information, depth maps, and segmentation maps ($\boldsymbol{m}_v$ and $\boldsymbol{m}_a$), 256 AMD EPYC 7763 CPU cores worked for 30 days.

A.1. Mass Estimation

Mass plays an important role in determining the outcome of object dynamics and interactions. While GSO provides a diverse collection of high-quality scanned 3D models for household items, physical properties such as mass and friction were not captured for many objects [25]. In Kubric *MOVi-F*, a constant density assumption is therefore made by default to estimate mass from volume [31]. In an attempt to increase the realism of our training data, we leverage GPT-3 [16] to produce rough estimates of the mass of every object in the GSO library based on its description and metadata. This is illustrated in Figure 8. In practice, we calculate and apply the geometric mean of the original and LLM-estimated mass, because the numbers provided by GPT-3 are, qualitative speaking, not always very accurate.

[6]This is the same as *MOVi-E*, but with a small degree of motion blur added to the video recorded by the virtual camera.

Figure 7. **Containers in GSO.** We mark roughly 11% of the assets in Google Scanned Objects [25] to be containers, which are spawned more often than average compared to other object types in Kubric Random.

B. Network Implementation Details

B.1. AOT

Since AOT is designed for VOS, we keep the entire pipeline of the AOT model intact for fairness. Following [73], at training time, a context window of 5 frames is fed into the model for a single training step, while at test time, the target object mask is propagated throughout the entire video clip from start to end.

B.2. TCOW

TCOW is a modification of the TimeSFormer network, which operates by processing a number of chunks of space-time patches into a transformer [11, 67]. Specifically, we concatenate the input video and the query mask along the channel axis to form $(\boldsymbol{x}, \boldsymbol{m}_q) \in \mathcal{R}^{T\times H\times W\times 4}$ (here, $\boldsymbol{m}_{q,t} = 0$ for all $t \geq 1$ as only the first frame is labeled). Similarly to Vision Transformer [22], the resulting set of frames is decomposed into $N = T \times h \times w$ small image patches of size $16 \times 16 \times 4$ each, with $h = \frac{H}{16}$, $w = \frac{W}{16}$. After a per-patch linear projection, an input sequence of

Category: *Consumer Goods*
Description: *Epson UltraChrome T0548 Ink Cartridge, Matte Black - 1-pack*
Height: *11.54 cm.*
[Mass / Weight]: *[0.065 kg. / 0.065 kg. / 0.09 kg. / 0.14 kg.] + random text*

Category: *None*
Description: *Vtech Roll & Learn Turtle*
Height: *11.75 cm.*
[Mass / Weight]: *[0.1 kg. / 0.09 kg. / 0.1 kg. / 0.5 kg.] + random text*

Figure 8. **Estimating mass for objects used in Kubric simulations.** We perform text completion with a large language model. Specifically, we query OpenAI GPT-3 (text-davinci-002) [16] twice for mass, twice for weight, and average the four numerical outputs after appropriate unit conversions. The image is shown for visualization only, and is not fed to the language model. The underlined text represents the four actual completion outputs made by GPT-3. The italic parts of the input are derived from the available metadata of each asset, and this procedure is repeated for all 1,032 GSO objects.

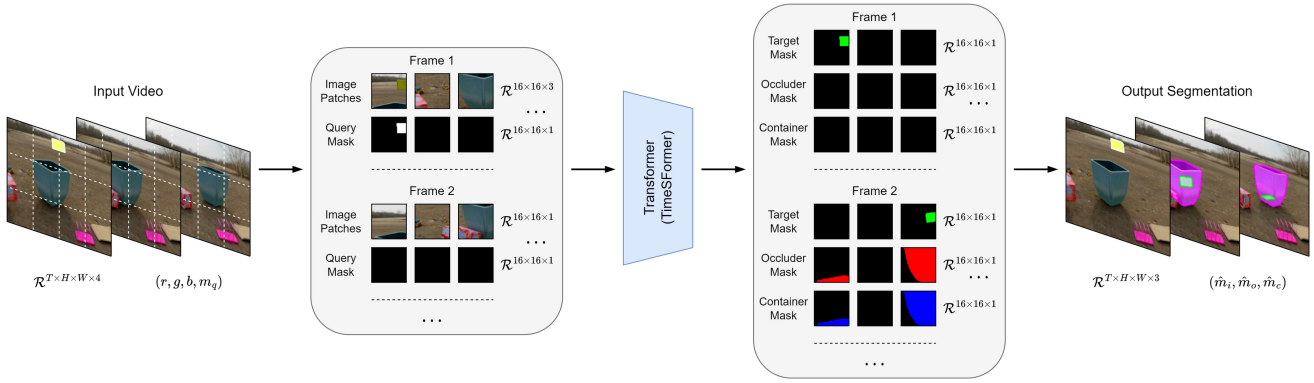


Figure 9. **TCOW architecture.** We apply the standard TimeSFormer backbone onto the input video $(\boldsymbol{x}, \boldsymbol{m}_q)$ following a spacetime divided attention scheme [11], but interpret the tokens after the transformer as patches for the predicted output masks. (Multiple channels belonging to the same patch are shown in separate tiles for clarity.)

N embeddings of dimensionality 768 is fed into a transformer, where we subsequently apply repeated multi-head self-attention blocks on these tokens.

The output sequence is treated as a spatiotemporal feature map for the purpose of dense video segmentation. Each element after the last attention layer is linearly projected back to image space, resulting in a set of patches of $16 \times 16 \times 3$, where the last dimension represents the predicted triplet of masks $(\hat{\boldsymbol{m}}_t, \hat{\boldsymbol{m}}_o, \hat{\boldsymbol{m}}_c)$. The vectors are composed in the same order as they were decomposed at the input side. The classification token is ignored and there is no pooling. A diagram is shown in Figure 9.

B.3. Learning and Supervision

We train the TCOW model for tracking objects through occlusion and containment by producing segmentation masks for each type. The network f (as defined in Equation 1) accepts a single query instance at a time, which makes a binary cross-entropy objective $\mathcal{L}_{BCE}$ between every output channel $\hat{\boldsymbol{m}}$ and its corresponding ground truth $\boldsymbol{m}$ a logical starting point.

Since the number of frames where the target is occluded is typically smaller than the number of frames where the target is visible in our training set, we scale $\mathcal{L}_{BCE}$ by a factor $1+(\beta-1)o$, where $o \in [0, 1]$ is the occlusion fraction.

However, inspired by [73], we also combine $\mathcal{L}_{BCE}$ with two additional loss terms: (1) a bootstrapped variant $\mathcal{L}_{BCE,k}$ that focuses on a certain top fraction k of pixels in each example that incur the highest individual contributions to the loss $\mathcal{L}_{BCE}$, and (2) a soft Jaccard loss $\mathcal{L}_{\mathcal{J}}$ [12]. The terms are linearly combined and weighted as follows:

$$\mathcal{L}_{\boldsymbol{m}} = (\lambda_1 \mathcal{L}_{BCE} + \lambda_2 \mathcal{L}_{BCE,k} + \lambda_3 \mathcal{L}_{\mathcal{J}})(\hat{\boldsymbol{m}}, \boldsymbol{m}) \quad (4)$$

Finally, the total objective is a weighted sum over the three different output types predicted by f:

$$\mathcal{L} = \lambda_t \mathcal{L}_{\boldsymbol{m}_t} + \lambda_o \mathcal{L}_{\boldsymbol{m}_o} + \lambda_c \mathcal{L}_{\boldsymbol{m}_c} \quad (5)$$

where $\mathcal{L}_{\boldsymbol{m}_t}$ addresses the target instance mask, $\mathcal{L}_{\boldsymbol{m}_o}$ is for the main occluder mask, and $\mathcal{L}_{\boldsymbol{m}_c}$ is for the main container mask. The ground truth masks for the latter two ($\boldsymbol{m}_o$ and

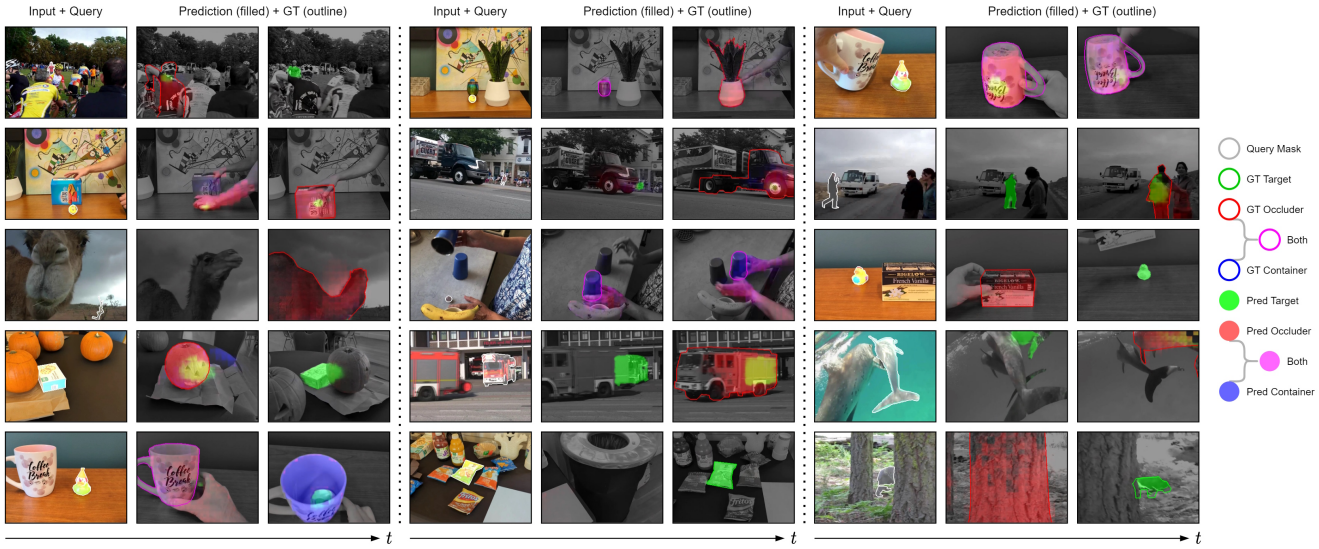


Figure 10. **Success cases for TCOW on Rubric.** All visualized predictions are made by the non-ablated TCOW network. This model performs particularly well on relatively simple cases of (total) occlusion and/or containment in the real world, despite being trained on synthetic data only. Some video clips with containers moving to a limited degree are also handled correctly (see middle center, or top right). However, more advanced examples of object permanence often result in failures, shown in Figure 13, demonstrating that a lot of room for improvement remains.

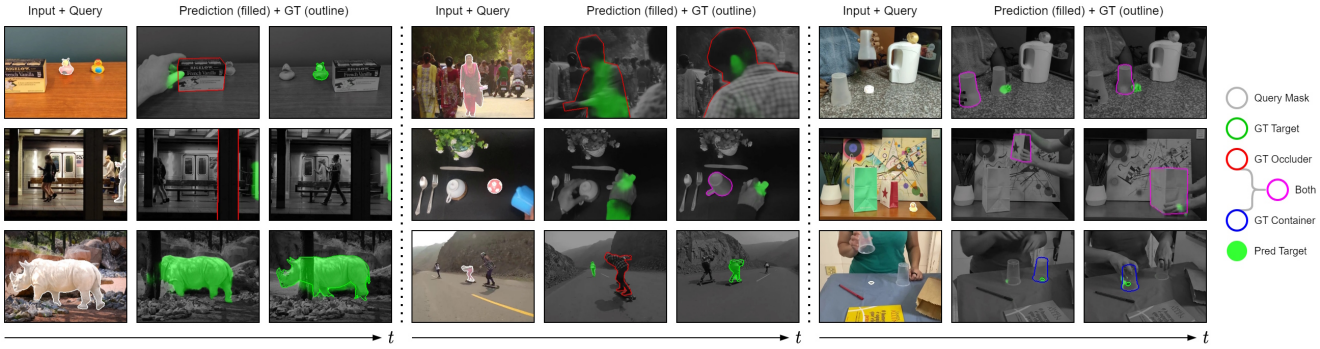


Figure 11. **Qualitative results for AOT on Rubric.** All visualized predictions are made by the non-ablated AOT network, and mirror Figure 6 in the main text. Although all models are trained on Kubric data with X-ray supervision, AOT often loses track as soon as total occlusion happens, and tends to jump to different instances or moving parts of the video (such as hands).

$\boldsymbol{m}_c$) are defined to be all-zero whenever there exists no occluder or container respectively, although for class balancing purposes, the loss is also weighted with a factor $\alpha < 1$ for those frames.

Augmentations during training consist of random color jittering (hue, saturation, brightness), random grayscale, random video reversal, random palindromes (*i.e.* playing clips forward and then backward, or vice versa), random horizontal flipping, and random cropping. We do not apply any augmentations at test time.

In Kubric Random, there are many possible objects with available annotations to track. At training time, we assign a difficulty score to every instance (that is visible in the first frame) based on its average occlusion fraction and how much motion it experiences over time. The query is then sampled randomly but non-uniformly, with preference given to the harder to track target objects. At test time, we measure and average metrics over the top four instances with the highest difficulty score per video. Other datasets (*i.e.* Kubric Containers plus all of Rubric) only have one designated target object per video clip.

In our experiments, we set $(T, H, W) = (30, 240, 320)$, $\beta = 5$, $(\lambda_1, \lambda_2, \lambda_3) = (0.2, 0.4, 0.4)$, $(\lambda_t, \lambda_o, \lambda_c) = (1.0, 0.5, 0.5)$, and $\alpha = 0.02$. The bootstrap fraction k is a function of time, and decreases linearly from 1 to 0.15 during the first 10% of training. We use the AdamW op-

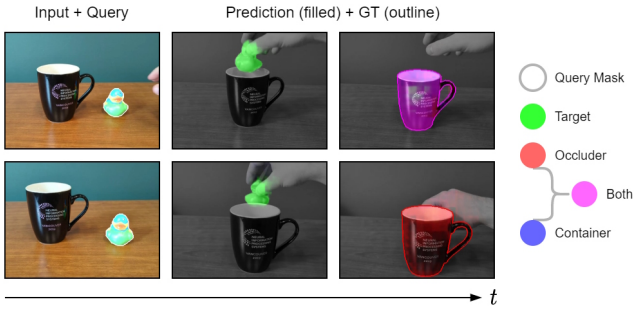


Figure 12. **Measuring TCOW's ability to differentiate occlusion from containment.** The first video is a control example where the duck is inserted *inside* the mug, such that it becomes simultaneously a container and occluder (magenta). In the second video, we pretend to do the same, but actually place it *behind* the mug, such that it becomes an occluder (red) only. Our TCOW model handles both cases correctly, suggesting that the learned representation is capable of spatial reasoning in a way that goes beyond just memorizing object class information (*e.g.* a container must contain an object whenever it hides one).

timizer and train for 70 epochs, which takes 3 days on 2 NVIDIA RTX A6000 GPUs. Inference (without gradients) happens in 0.27 seconds for a single clip, which corresponds to roughly 110 FPS.

C. More Qualitative Results

Please see Figures 10, 11, and 13, as well as tcow.cs.columbia.edu for videos along with explanations. We recommend viewing the project webpage in a modern browser.

C.1. Differentiating containers from occluders

Distinguishing occlusion from containment can be challenging, especially if a potential container is itself responsible for merely occluding but not containing a target object. One aspect of our TCOW Rubric Office benchmark therefore analyses the interesting scenario where we attempt to trick the model into confusing containment with occlusion. We evaluate this in Figure 12, which illustrates that the TCOW network capitalizes on motion cues, and not (only) object category information.

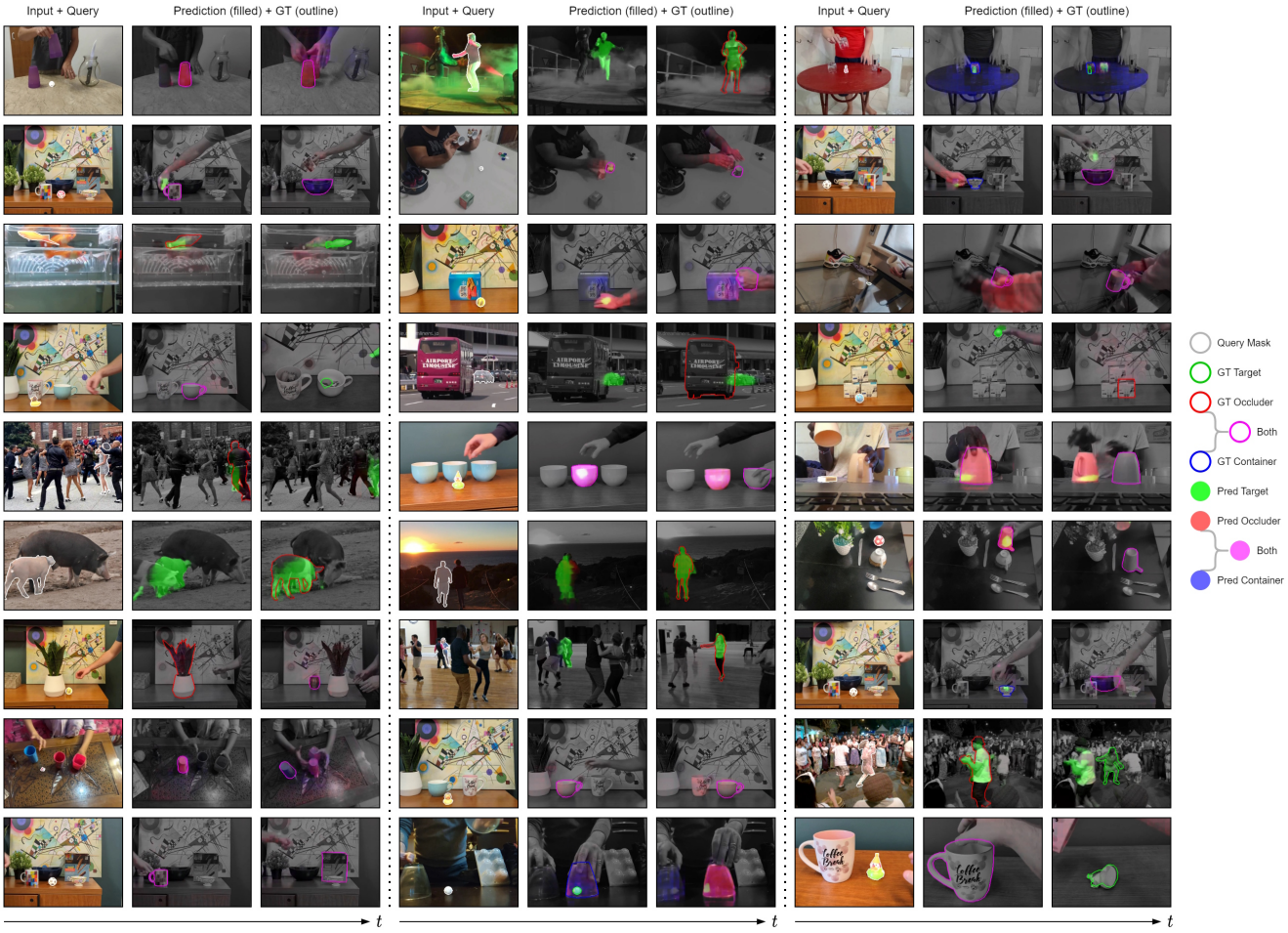


Figure 13. **Failure cases for TCOW on Rubric.** All visualized predictions are made by the non-ablated TCOW network. Multiple trends could be discerned among real-world scenarios where the model fails, which can roughly be summarized as: (1) identical containers, one of which is holding the target object, being shuffled around; (2) nested containment, *e.g.* when a mug is placed inside a larger box; (3) the occluder and occludee are visually very similar, *e.g.* people occluding people or animals occluding animals. By releasing this challenging benchmark to the community, we hope future work will be able to address these cases more successfully.