Open-World Object Counting in Videos

Niki Amini-Naieni, Andrew Zisserman

Visual Geometry Group, Dept. of Engineering Science, University of Oxford, UK
nikian@robots.ox.ac.uk, az@robots.ox.ac.uk

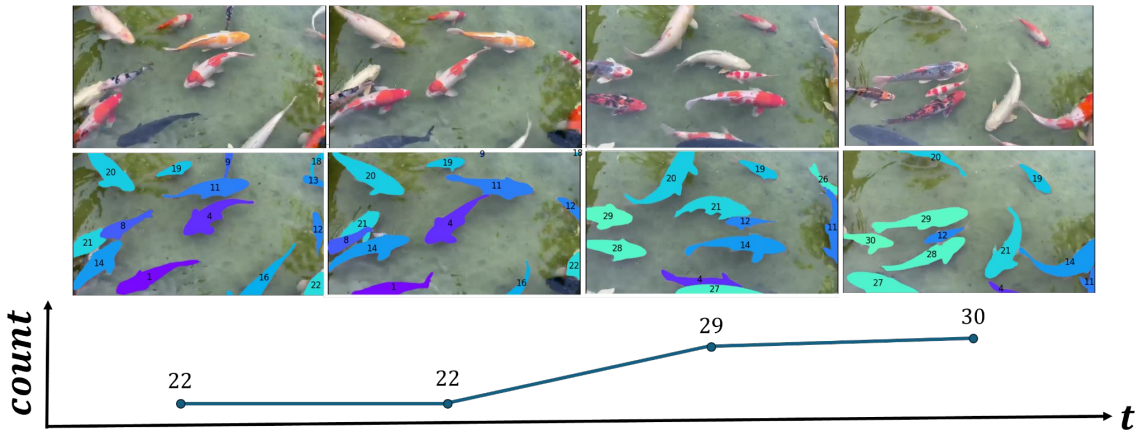

Figure 1: Object counting in a video. Given the video in the top row (shown by a few sample frames) and the text "fish", COUNTVID is able to accurately match instances across the video (indicated by the color and number assigned to each fish), identify new objects, and estimate the count. The video is available on the project website.

Abstract

We introduce a new task of open-world object counting in videos: given a text description, or an image example, that specifies the target object, the objective is to enumerate all the unique instances of the target objects in the video. This task is especially challenging in crowded scenes with occlusions and objects of similar appearance, where avoiding double counting and identifying reappearances is crucial. To this end, we make the following contributions: we introduce a model, COUNTVID, for this task. It leverages an image-based counting model, and a promptable video segmentation and tracking model, to enable automated open-world object counting across video frames. To evaluate its performance, we introduce VIDEOCOUNT, a new dataset for this novel task built from the TAO and MOT20 tracking datasets, as well as from videos of penguins and metal alloy crystallization captured by x-rays. Using this dataset, we demonstrate that COUNTVID provides accurate object counts, and significantly outperforms strong baselines[1].

Introduction

Our objective in this paper is *open-world* object counting in videos – determining how many instances of an object class are present in a video, where the object class of interest is specified by a text description or an image exemplar and may not have been included in the training data. This is a time-dependent task, as both the current (visible count) can be reported at the frame level, or other temporal intervals, as well as the cumulative count over the entire video.

As fig. 1 illustrates, keeping count in videos is naturally a *correspondence or tracking task* – since we do not want to count the same instance multiple times, we must establish that instances in consecutive frames are the same. However, as the figure also illustrates, one of the fundamental challenges of counting in videos is *instance identification* – is an

[1]COUNTVID code and the VIDEOCOUNT dataset are available at https://www.robots.ox.ac.uk/vgg/research/countvid/.

object appearing in a frame a new instance? Or, is it one that drifted out of frame earlier in the sequence or was temporarily occluded? This challenge is exacerbated as the objects become indistinguishable: discriminating individual fish is possible if they have different colors and markings, but discriminating insects or crows may be impossible.

Somewhat surprisingly, automated counting in *videos* is a relatively unexplored area. While there are a few automated methods for closed-vocabulary counting (Loy et al. 2013; Fang et al. 2019; Zhu et al. 2021a; Makhura and Woods 2019; Han et al. 2022; Wen et al. 2021; Liu et al. 2024b), there are no open-vocabulary methods that we are aware of. This in contrast to the extensive exploration of counting in *images*, where open-vocabulary methods are able to use both text and exemplars to specify the target object, and can count up to thousands of instances (Amini-Naieni, Han, and Zisserman 2024; Pelhan et al. 2024a; Ranjan et al. 2021). Even some large-scale Vision-Language Models, such as Molmo (Deitke et al. 2025), are now able to accurately count beyond ten instances in an image.

This lack of research in video counting is especially surprising given the wide variety of science applications and need. Conservationists need to count animals in video sequences captured by drones for population monitoring (Jones et al. 2023; Mustafa et al. 2019). This can take up to 30 hours for a trained human analyst to annotate manually for a single one-hour flight (Wich et al. 2021). Material scientists count crystals forming from liquid metal alloys to determine how cooling affects the speed of the formation process (Liotti et al. 2018). Epidemiologists use human and vehicle counts from videos captured on city streets to study causes of pedestrian exposure to air pollution and mitigate them (Schroeder et al. 2024, 2020). An 'open-world' method that can be quickly applied to all these problems out-of-the-box, with no manual counting or additional training needed, has the potential to catalyze these applications, eliminating the annotation time, and significantly benefiting their research.

In this paper, we introduce a model, COUNTVID, for open-world object counting in videos that accepts a video and a prompt that specifies the target object to count as inputs, and outputs how many unique instances of the object appear in the video. The prompt can consist of a free-form text description and/or any number of 'visual exemplars', where the visual exemplars indicate the object of interest by bounding boxes and can come from a video frame or an external image.

The COUNTVID model builds on advances for two distinct tasks: (i) powerful open-vocabulary image counting and detection models (Amini-Naieni, Han, and Zisserman 2024; Pelhan et al. 2024b; Zhizhong et al. 2024); and (ii) powerful class-agnostic video segmentation and tracking models (Ravi et al. 2025; Yang et al. 2024). However, naively combining an image detector to provide instances for a tracker is not sufficient since state-of-the-art object detectors struggle to count high numbers of objects in densely packed scenes with many occlusions and overlapping instances (Amini-Naieni, Han, and Zisserman 2024; Dai, Liu, and Cheung 2024). To overcome this detection problem, for our model we leverage accurate image-based *counters* that also output the bounding boxes required by the tracker (Pelhan et al. 2024b; Zhizhong et al. 2024).

One of the innovations of this paper is to extend the most flexible image-based counter, CountGD (Amini-Naieni, Han, and Zisserman 2024), to also produce bounding boxes, offering versatility in accepting text, visual exemplar, or combined prompts and detection capabilities within the same model. We show that its performance surpasses image detectors when there are many instances of similar objects in the image, and also prior models (Amini-Naieni, Han, and Zisserman 2024; Pelhan et al. 2024a; Nguyen et al. 2022).

A second innovation is to propose a temporal filter to remove false positive tracks resulting from erroneous detections. The extended CountGD model, dubbed COUNTGD-BOX, and other detection-based counters are used to provide box prompts over multiple frames in the video, and the tracker is used to *associate* the resultant segmentations and propagate them to other frames.

To assess the performance of video counting, we introduce VIDEOCOUNT, a new dataset with ground truth for this task. VIDEOCOUNT has two types of benchmarks: first, we re-purpose standard tracking datasets, TAO (Dave et al. 2020) and MOT20 (Dendorfer et al. 2020), by adding additional annotations to ensure all objects are counted (since tracking benchmarks typically only evaluate on a subset of the objects, e.g. not accounting for static objects); and second, we introduce two science applications of counting with new videos containing real footage from monitoring penguins in their natural habitats and x-ray images of the crystallization process for metal alloys. The number of objects in videos from our dataset ranges from one to over a thousand.

In summary, we make the following four contributions: *first* we present the novel task of open-world object counting in videos; *second* we propose a model for this task, COUNTVID, by re-purposing and combining open-vocabulary image counting and class-agnostic segmentation and tracking models; *third* we extend CountGD to produce bounding boxes as outputs, and introduce an automated method to remove false tracks; and *fourth* we release VIDEOCOUNT, a new video dataset for evaluating algorithms for this open-world object counting task.

Related Work

A note about terminology: in the object counting literature (Amini-Naieni et al. 2023; Amini-Naieni, Han, and Zisserman 2024; Liu et al. 2022), *open-world counting* refers to counting instances of an object class specified at test time via text or visual prompts. We adopt this definition, but also discuss how 'open-world' has been interpreted differently in the past in the appendix.

Open-World Object Counting in Images. Prior work on open-world object counting only focuses on images. The first image-based methods required the user to manually annotate a few example objects with 'visual exemplars' to count at inference time (Liu et al. 2022; Lu, Xie, and Zisserman 2018; Nguyen et al. 2022; Ranjan et al. 2021; Shi et al. 2022; Yang et al. 2021; You et al. 2023; Djukic et al. 2023;

Lin et al. 2022). More recent works (Dai, Liu, and Cheung 2024; Amini-Naieni et al. 2023; Amini-Naieni, Han, and Zisserman 2024; Jiang, Liu, and Chen 2023; Kang et al. 2024) have leveraged pre-trained vision-language foundation models to enable the category to be specified by text. CountGD is a state-of-the-art open-world counting model that uses the joint vision-language embedding space of the Grounding DINO (Liu et al. 2024a) foundation model to allow the user to specify the object to count with text. In addition to this, unlike most prior approaches that either only accept text or only accept visual exemplars, CountGD allows for *both* inputs. By accepting only text, CountGD can adapt to novel classes without human intervention, and by accepting visual exemplars, it provides greater accuracy. We build on CountGD in this work.

Open-World Object Counting in Videos. While there is no prior work that explicitly focuses on open-world object counting in videos, there are open-world trackers (Li et al. 2023) that can be repurposed as counters. State-of-the-art open-world trackers rely on object detectors (Li et al. 2024b,a; Qian et al. 2024; Heigold et al. 2023). For example, the open-world tracker MASA (Li et al. 2024a) leverages detectors such as Grounding DINO and Detic (Zhou et al. 2022) to first detect any object using text and then associate it across video frames. The unique objects identified and tracked throughout the video can be enumerated to estimate the count. VOVTrack (Qian et al. 2024) uses region proposals from Faster R-CNN (Ren et al. 2017) for object localization. Because these approaches extend detectors, they inherit their limitations.

Trackers that do not rely on pre-trained image-based object detectors also have limitations. Trackformer (Meinhardt et al. 2022) cannot adapt to novel categories at inference, only tracking objects that it has been trained to track. On the other hand, OVTR (Li et al. 2025) is an end-to-end transformer-based tracking model that can track novel objects given text input. However, it does not accept visual prompts limiting its accuracy. Furthermore, it has not been trained or tested on scenes with hundreds to over a thousand objects. SAM 2, SAM 2.1 (Ravi et al. 2025) and SAMURAI (Yang et al. 2024) are recent state-of-the-art tracking and segmentation models that also adapt to novel objects without retraining. SAM 2.1 and SAMURAI extend SAM 2 with motion cues, longer video training, and occlusion handling, but both require manual prompting, which COUNTGD-BOX and COUNTVID automate. SAMURAI also focuses mainly on single-object tracking.

COUNTVID and COUNTGD-BOX Models

In this section, we first present COUNTVID, our method for open-world object counting in videos. We then present COUNTGD-BOX, a multi-modal counting model that outputs bounding boxes and extends CountGD.

COUNTVID

COUNTVID is a model that inputs a video and flexible prompts including text only, visual exemplars only, or both and outputs frame-level counts and a global count indicating the number of unique objects in the video that match the prompts. At inference, COUNTVID processes the video in three stages, at decreasing levels of granularity. The stages are illustrated in fig. 2.

Stage 1 – Processing at the Frame Level. The first stage is where the visual exemplar and text prompts are used, and the objective is to automatically obtain bounding boxes and segmentation masks for all instances of the target object in each frame. To achieve this, the visual exemplar and text prompts are fed to a counting and detection model that is applied to each video frame independently to obtain bounding boxes. The bounding boxes output by the counting model are used as box prompts for a segmentation model, which outputs masks for all the objects in the frame. We use exemplars from a single frame and apply them across the video to reduce user annotation effort. To improve efficiency, frames are subsampled before this stage begins.

Stage 2 – Processing in the Short Term. Although counting models are very accurate at counting, they can still produce false positive detections. This may occur due to motion blur. The objective of Stage 2 is to remove these false positive predictions with a temporal filter. This filter leverages the observation that false positives tend to be transient in independent (per-frame) predictions from Stage 1, disappearing almost immediately in subsequent frames. For each detection from Stage 1, the filter checks if the object exists within a temporal window of w frames. Using a segmentation and tracking model, the filter tracks $w-1$ frames backwards and $w-1$ frames forward from the frame the detection is in, resulting in a span of $2w-1$ frames. Objects are matched using the intersection over union (IoU) of the propagated masks from tracking and the masks from independent per-frame detections in Stage 1. An IoU greater than 0.5 is considered a match. If the object is matched in a sequence of at least w consecutive frames, it is kept. This ensures the filter tolerates a degree of occlusion or intermittent visibility. Otherwise, it is removed before Stage 3 begins. Note, it is necessary to track both forwards and backwards in time, as new objects can appear (and are verified by tracking forwards in time), and also objects can disappear (e.g. by occlusion) and are verified by tracking backwards in time.

Stage 3 – Processing in the Long Term. In the last stage, COUNTVID applies a segmentation and tracking model to the full video, keeping track of objects in the long term while checking for new objects in each frame. For each object, COUNTVID predicts a *masklet*, an object mask propagated over time. New objects are detected by comparing existing masklets with per-frame masks from Stage 2. Per-frame masks that do not overlap with the existing masklets are identified as new objects. The new objects are then also tracked going forward. Once all the frames have been checked for new objects, the masklets are enumerated to estimate the final global count. The New Object Detection Logic is visually explained in detail in the appendix.

COUNTVID Implementation. We implement COUNTVID with COUNTGD-BOX (described below) as the counting/detection model, and SAM 2.1 as the tracker. The boxes from COUNTGD-BOX are used as box prompts for SAM 2.1

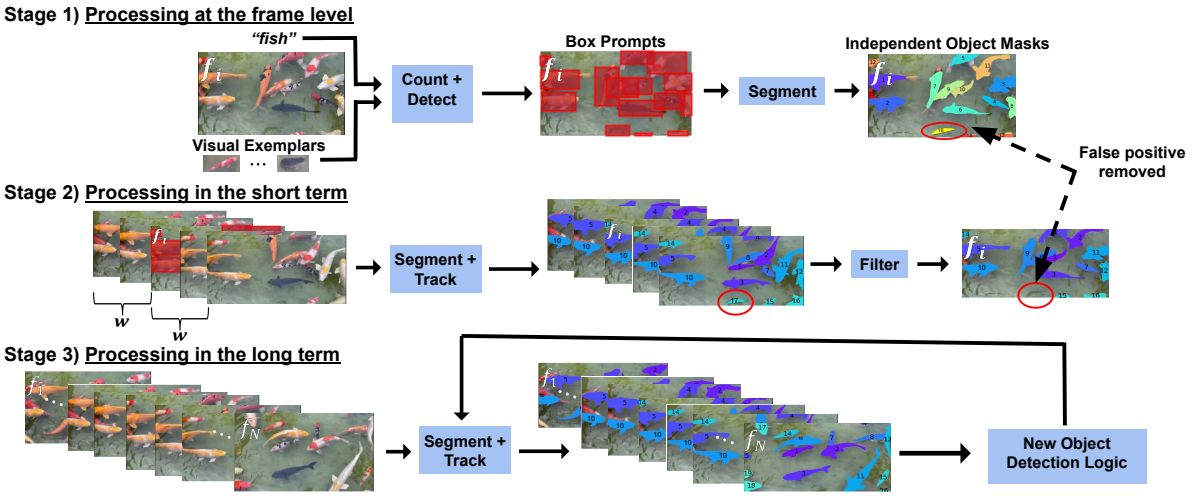

Figure 2: Inference with COUNTVID. Each stage processes the video at decreasing granularity. In Stage 1, a detection-based counter uses text and exemplars to generate box prompts for each frame, which are used by a segmentation model to produce object masks. Stage 2 applies a temporal filter over sequences of w frames to remove transient false positives (e.g., the red-circled object in f_i). Stage 3 propagates object masks from Stage 2 across the full video while checking for new objects in each frame. All the identified objects are enumerated to produce the final global count.

in Stage 1, and then SAM 2.1 tracks the masks from the prompted objects, producing masklets. Though COUNTVID is agnostic to the choice of counter/detector and tracker. We compare to other choices in the experiments.

COUNTGD-BOX

To automatically obtain box prompts for the segmentation model, we need a detector that can handle dense scenes with many similar overlapping objects, because this will occur in the video for our challenging task. As shown by our results and prior work (Amini-Naieni, Han, and Zisserman 2024; Amini-Naieni et al. 2023), there are open-world detectors, but they do not perform well in this setting. On the other hand, there are open-world counters that do. Out of all these counters, CountGD is the most flexible, accepting either text only, visual exemplars only, or both simultaneously to specify the object. It also provides generally strong counting performance across all prompt settings as shown by our results. However, unlike other less flexible detection-based counting models, CountGD outputs points, not boxes.

Point prompts do not specify the object to count in a well-defined way. For example, a point on the window of a car could mean every window or every car should be counted. Given there is likely more than one window on a car, this problem could result in an erroneously high or low count. This ambiguity is resolved if an image region (a box) is specified instead of a point.

To obtain well-defined object prompts for the segmentation model, we train CountGD to output *boxes* in addition to *points*. CountGD originally lacks this capability due to the limited bounding box data available in object counting datasets such as FSC-147 (Ranjan et al. 2021). This scarcity exists because labeling hundreds to thousands of objects with bounding boxes is extremely tedious. Instead, these datasets only provide box annotations for a few objects per image. Drawing inspiration from DAVE (Pelhan et al. 2024a), we extend CountGD to take advantage of these weak training labels. Because CountGD is built on top of the Grounding DINO architecture, it already outputs four parameters per object. The first two are used as the center of the object, while the last two are discarded by CountGD. We add two new terms, $\mathcal{L}^e_{h,w}$ and $\mathcal{L}^e_{GIoU}$, to CountGD's loss, to train the last two parameters to be the height and width of the bounding box. The new loss is defined as $\mathcal{L} = \lambda_{loc}\left(\mathcal{L}^e_{h,w} + \mathcal{L}_{center}\right) + \lambda_{GIoU}\mathcal{L}^e_{GIoU} + \lambda_{cls}\mathcal{L}_{cls}$ where $\mathcal{L}^e_{h,w}$ and $\mathcal{L}^e_{GIoU}$ are based on the bounding box regression losses from Grounding DINO. The difference here is that these losses are only calculated for the exemplars, while in Grounding DINO, they are calculated for all the objects in the image. $\mathcal{L}^e_{h,w}$ is the sum of the absolute errors of the height and widths and $\mathcal{L}^e_{GIoU}$ is the generalized intersection over union between the predicted and ground truth exemplar boxes. By training on the exemplar boxes, CountGD learns to not only predict points, but to also predict bounding boxes. We name the extended CountGD as COUNTGD-BOX and use it at inference to produce box prompts for the segmentation model.

VIDEOCOUNT: A New Counting Dataset

Current benchmarks for object counting are not sufficient for the open-world object counting in videos task. This is because existing counting datasets either only support images (Ranjan et al. 2021; Hsieh, Lin, and Hsu 2017) or only include a limited number of categories (Loy et al. 2013; Zhu et al. 2021b; Dendorfer et al. 2020). Furthermore, ex-

	# Videos	# Classes	# Objects	Video Len
TAO-Count	357	139	1-10 2.7 Avg	8s-78s 34s Avg
MOT20-Count	3	1	80-1203 520.3 Avg	17s-133s 87s Avg
Science-Count	10	2	10-154 63 Avg	4s-30s 11s Avg

Table 1: VIDEOCOUNT details. Our dataset, composed of 3 benchmarks, covers a wide range of object categories (141 in total) and a wide range of object counts (1-1203 per video).

isting tracking datasets such as TAO (Dave et al. 2020) only provide exhaustive annotations for a subset of objects and only label at most ten objects per video, which is far too low for practical counting use cases. Therefore, in this section, we present VIDEOCOUNT, a new dataset for open-world object counting in videos that overcomes these limitations. VIDEOCOUNT is made up of three benchmarks: TAO-Count, MOT20-Count, and Science-Count. It contains 370 videos covering a wide range of object categories and counts as shown in table 1. We include further details in the appendix.

Our dataset is built from diverse sources. For TAO-Count and MOT20-Count, we add metadata to subsets of the existing tracking datasets TAO (Dave et al. 2020) and MOT20 (Dendorfer et al. 2020) specifying the counts of target objects. For Science-Count, we release new videos and count annotations from monitoring penguin populations and videoing the formation of crystals from liquid metal alloys captured with x-ray radiography (Liotti et al. 2018). Example video frames from VIDEOCOUNT are shown in fig. 3 and the appendix.

VIDEOCOUNT tests COUNTVID's ability to adapt to a wide range of challenging scenarios. TAO-Count tests how well COUNTVID counts low numbers of objects in scenes with significant motion. MOT20-Count tests how well COUNTVID counts in heavily crowded scenes (e.g., > 1000 objects) with many overlapping instances. Science-Count evaluates COUNTVID on tricky real-world applications with many similar objects, some even rapidly changing structure over time in x-ray videos typically out-of-domain for foundation models.

Experiments

Implementation Details: COUNTGD-BOX is initialized with the pre-trained weights of CountGD. Its MLP box detection heads are then fine-tuned on the FSC-147 training set for 30 epochs with early stopping on the validation set. λ_{loc}, λ_{GIoU}, and λ_{cls} in the loss are set to 5, 2, and 2 respectively using a grid search on the validation set. For COUNTVID, in Stage 1, we sample frames at 3 fps. In Stage 2, the window size of the temporal filter, w, is set to 3 frames, corresponding to one second. The IoU threshold for matching is set to 0.5. Additional implementation details, including detailed analysis of the inference time and memory consumption for each stage with and without the temporal filter, are given in the appendix.

Datasets & Metrics

Images: To evaluate the counting and detection accuracy of state-of-the-art detection and counting models in crowded scenes, we use FSCD-147 (Nguyen et al. 2022), which provides bounding boxes for the validation and test sets of the widely established open-world image object counting dataset FSC-147. Exhaustive bounding boxes are not provided for the training set. Each image is annotated with three visual exemplars. To measure the detection accuracy, we follow (Nguyen et al. 2022) by providing the mean average precision over thresholds 0.5 to 0.95 (AP) and the average precision at the IoU threshold of 0.5 (AP50). We also report the image-based count MAE and RMSE used in (Ranjan et al. 2021). We take the count as the enumeration of the bounding boxes following (Pelhan et al. 2024b), and report results given text only, three exemplars only, or both, depending on what each method allows. In addition, we report zero-shot counting results for detectors and counters on the ShanghaiTech (Zhang et al. 2016) crowd counting dataset in the appendix.

Videos: To evaluate counting accuracy for videos, we report results on VIDEOCOUNT. Due to the class overlap of the training set of FSC-147 and TAO-Count, we also report results on the subset of videos in TAO-Count with the training classes in FSC-147 removed. To measure counting accuracy for videos, we draw on prior work on object counting in images (Ranjan et al. 2021) that uses the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). We define video analogues of these metrics for our new task. More specifically, we define the video MAE and RMSE as: $MAE = (1/N)\sum_{i=1}^{N}|\hat{y}_i - y_i|, RMSE = [(1/N)\sum_{i=1}^{N}(\hat{y}_i - y_i)^2]^{1/2}$, where N is the number of test videos, $\hat{y}_i$ is the predicted count for video X_i, and y_i is the ground truth count for video X_i. In more detail, y_i is the number of *unique* objects in the video that match the prompts. Importantly, the MAE and RMSE metrics for counting in videos differ from those used for images. In the video setting, the ground truth count reflects the number of *unique object identities*, not the number of detections. This requires both matching and re-identification: objects that reappear across frames must not be double-counted, so repeated detections of the same object must be correctly associated.

Assessing Processing at the Frame Level

In table 3, we evaluate different *image* counting and detection methods on FSCD-147 Test given different prompts including text only ('text'), exemplars only ('exemp'), or both together ('both'). Results for FSCD-147 Val are given in the appendix. Following (Pelhan et al. 2024b), the count is determined by enumerating the bounding boxes, not by summing density maps (Pelhan et al. 2024a) or applying test-time procedures (Amini-Naieni, Han, and Zisserman 2024) as these operations do not provide boxes. For exemplars, we use the three provided by FSC-147 for each sample. Text descriptions either come from the FSC-147 class names or

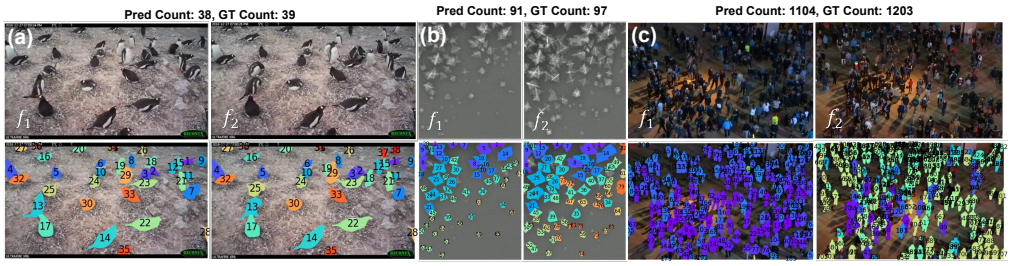

Figure 3: Qualitative results on VIDEOCOUNT. f_1 and f_2 are frames sampled consecutively in time. COUNTVID handles dense (b, c), deforming (b), and similar (a, b) objects.

	TAO-Count		TAO-Count-FSC		MOT20-Count		Penguins		Crystals	
Method	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
DR. VIC (Baseline)	-	-	-	-	100.0	110.8	-	-	-	-
MASA (Baseline)	14.1	21.5	13.9	20.4	630.0	725.4	9.0	11.5	72.3	87.9
COUNTVID (Ours)	**2.6**	**6.0**	**2.5**	**5.2**	**50.0**	**61.3**	**4.0**	**5.3**	**69.1**	**86.0**

Table 2: Counting performance on VIDEOCOUNT. MASA and COUNTVID are given *only text* prompts. DR. VIC only counts humans and so can only be evaluated on MOT20-Count. TAO-Count-FSC is TAO-Count with the training classes in FSC-147 removed. The class overlap between TAO-Count and FSC-147 has no significant influence on COUNTVID's counting accuracy.

		FSCD-147 Test			
		Counting		Detection	
Method	Prompt	MAE ↓	RMSE ↓	AP ↑	AP50 ↑
GDINO	text	54.16	157.87	11.60	17.80
OWLv2	text	41.83	149.82	22.84	35.76
PSeCo	text	16.58	129.77	**41.14**	**69.03**
DAVE$_{prm}$	text	15.52	**114.10**	18.50	50.24
CountGD	text	15.19	119.40	18.10	52.90
CGD-B	text	**15.01**	118.16	30.44	61.56
C-DETR	exemp.	16.79	123.56	22.66	50.57
PSeCo	exemp.	13.05	112.86	**43.53**	74.64
DAVE	exemp.	10.45	74.51	26.81	62.82
GeCo	exemp.	**7.91**	**54.28**	43.42	**75.06**
CountGD	exemp.	10.77	99.51	19.76	57.24
CGD-B	exemp.	10.85	99.60	34.81	69.46
CountGD	both	**10.18**	**96.20**	20.50	59.40
CGD-B	both	10.29	96.33	**36.20**	**72.39**

Table 3: Results on FSCD-147 for image counting methods that output boxes (for Stage 1). The abbreviations are: COUNTGD-BOX (CGD-B); C-DETR (Counting-DETR (Nguyen et al. 2022)). SAM masks are not used here.

Temporal Filter	Prompt	MAE ↓	RMSE ↓
✗	text	6.6	17.5
✓	text	**2.6**	**6.0**

Table 4: Temporal filter ablation on TAO-Count (Stage 2).

FSC-147-D (Amini-Naieni et al. 2023). Note, we obtain box predictions from the original CountGD (without the additional training losses added in this paper) by using its full bounding box outputs, rather than, as in the original model, only using the points as output.

From these results, we draw three conclusions: (i) as confirmed by prior work, SoTA detectors like OWLv2 (Minderer, Gritsenko, and Houlsby 2023) and Grounding DINO (Liu et al. 2024a) do not work well for the counting setting, where there are many similar and overlapping objects. The caveat here is that these detectors have not been trained on FSC-147, as the counters have. To address this, we also benchmark OWLv2 and Grounding DINO against COUNTGD-BOX and CLIP-Count (Jiang, Liu, and Chen 2023) on the held-out ShanghaiTech counting dataset without any training on ShanghaiTech and show that the counters still significantly beat the detectors in the appendix; (ii) Extending CountGD to COUNTGD-BOX, significantly improves its detection accuracy while preserving its counting accuracy; (iii) the SoTA model depends on the type of prompt (text/exemplar/both) used. While COUNTGD-BOX is 'a good all-rounder,' it is not the best for all cases. Both COUNTGD-BOX and PSeCo (Zhizhong et al. 2024) perform competitively in the text-only setting. GeCo (Pelhan et al. 2024b) is the superior model in the exemplar-only setting, although both COUNTGD-BOX and DAVE (Pelhan et al. 2024a) are strong contendors. For models that accept both exemplars and text, COUNTGD-BOX is superior over CountGD for detection. In some cases, the text does add information to the exemplar, by specifying location or color for example (see section 4.5 of (Amini-Naieni, Han, and Zisserman 2024)). However, in settings where this is not the case, like FSC-147 where the text and exemplar both represent the class, GeCo given only exemplars should be used.

CountVid Variation	Prompt	Penguins		Crystals	
		MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
CGD-B/BT	text	4.3	5.5	71.6	88.1
CGD-B/S2	text	**3.3**	**4.8**	**69.1**	**86.0**
CGD-B/S2.1	text	4.0	5.3	**69.1**	**86.0**
CGD-B/BT	exemp	4.0	**4.2**	**31.1**	**52.8**
GeCo/S2	exemp.	11.7	13.1	59.6	104.2
GeCo/S2.1	exemp.	11.3	14.5	46.1	82.8
CGD-B/S2	exemp.	**3.3**	4.8	37.4	61.0
CGD-B/S2.1	exemp.	**3.3**	4.8	33.7	59.8
CGD-B/BT	both	7.0	8.3	12.7	16.6
CGD-B/S2	both	**0.3**	**0.6**	**12.0**	**13.5**
CGD-B/S2.1	both	0.7	0.8	13.4	14.8

Table 5: CountVid video counting results on Science-Count using various prompts, counters, and trackers. The abbreviations are: CountGD-Box (CGD-B); ByteTrack (BT); SAM 2 (S 2); and SAM 2.1 (S 2.1).

Assessing processing in the Short Term

In table 4, we assess the effectiveness of the temporal filter on TAO-Count. Specifically, we report the video-based MAE and RMSE from applying CountVid with and without the temporal filter given text only. For the counter, we use CountGD-Box, and for the tracker, we use SAM 2.1. The scenes in TAO-Count involve significant motion and blur, inducing false positives. The temporal filter effectively removes these false positives, reducing the MAE and RMSE by over 50%, improving the counting accuracy significantly.

Assessing Processing in the Long Term

In this section, we evaluate CountVid's overall video-based counting performance on VideoCount, and compare its performance to strong baselines. For the text descriptions for TAO-Count, we use the category synsets (Dave et al. 2020). We use the text 'human' for MOT20-Count. We use 'white crystal' and 'penguin' for Science-Count. When exemplars are used, 3–6 exemplars are provided for the first frame of the video and applied to all subsequent frames. The overall results are given in table 2 and table 5.

Baselines: We compare CountVid to strong baselines built from a Multi-Object Tracking (MOT) method and a Video Individual Counting (VIC) method. Specifically, for the first baseline, we repurpose the strong open-world tracker MASA (Li et al. 2024a) implemented with Grounding DINO. The unique tracks are enumerated to estimate the count. For the second baseline, we use the specialized crowd counting method DR. VIC (Han et al. 2022) trained on HT21 (Sundararaman et al. 2021) and only report results on MOT20-Count, as the other benchmarks in VideoCount contain objects other than humans. Note the HT21 training set contains two of the videos in MOT20-Count.

CountVid implemented with CountGD-Box and SAM 2.1 achieves significantly better counting accuracy than the MASA baseline, as shown in table 2. For a fair comparison with MASA, CountVid is given text only, even though CountVid can also accept exemplars. Remarkably, CountVid also significantly improves on the performance of DR. VIC, even though DR. VIC was trained specifically for counting humans and has been trained on one out of the three videos in MOT20-Count, so it has been trained on one third of the videos we are evaluating it on, while CountVid can count other objects in addition to humans and is applied to MOT20-Count zero-shot.

Note that none of the components of CountVid (including CountGD-Box and any hyperparameters) were fine-tuned on the data in VideoCount. We hypothesize the significantly superior performance of CountVid can be attributed to (i) leveraging the image-based *counting* model CountGD-Box rather than the Grounding DINO *detection* model to handle crowded scenes. CountGD-Box extends Grounding DINO by fine-tuning on counting data and adding modules to enable visual exemplar inputs, meaning it is more accurate and capable than Grounding DINO at counting in video frames; (ii) removing false positives with a temporal filter; and (iii) effectively leveraging video foundation models like SAM 2.1.

In table 5, we compare different variations of CountVid implemented with different combinations of counters/trackers and given different prompts. For the counters, we use CountGD-Box and GeCo, and for the trackers, we use ByteTrack (Zhang et al. 2022), SAM 2 and SAM 2.1. For the ByteTrack variant, we rely on the tracker to detect new objects and do not apply the temporal filter, since there are no segmentation masks. We note the exemplar-only performance is better than text-only, and providing both prompts is the best, showing CountVid effectively benefits from more information about the object. We find that while GeCo works well for images, it is not as accurate as CountGD-Box on videos. SAM 2.1 performs significantly better than SAM 2 in the exemplar-only setting for Crystals. However, it falls slightly behind SAM 2 for the other settings.

As shown in fig. 3, CountVid counts in dense scenes, detects new objects while retaining old ones, and counts deforming objects. Errors can occur due to false negatives from the counting model and tracker re-identification challenges. Scenes with many occlusions and similar instances can cause higher errors due to more of these cases.

Conclusion

We present the novel task of open-world object counting in videos together with a new model, CountVid, and a new dataset, VideoCount, to test the model. CountVid inputs flexible visual exemplar and text prompts and outputs both frame-level counts and a global count indicating the number of unique objects in the video that match the prompts. CountVid will continue to benefit from better trackers and class-agnostic detection-based counting models, as they can easily be plugged into the framework we have proposed.

Acknowledgments

The authors would like to thank Dr Tom Hart and Penguin Watch for the videos in the Penguins (Science-Count)

benchmark, Dr Enzo Liotti for the videos in the Crystals (ScienceCount) benchmark and Shun Yang for the curation and preparation of these videos, Jer Pelhan for his extensive support of GeCo, and Siyuan Li for his extensive support of MASA. This research is funded by an AWS Studentship, the Reuben Foundation, a Qualcomm Innovation Fellowship (mentors: Dr Farhad Zanjani and Dr Davide Abati), the AIMS CDT program at the University of Oxford, EPSRC Programme Grant VisualAI EP/T028572/1, and a Royal Society Research Professorship RSRP\R\241003.

References

Abhijit, B.; and Terrance, B. 2015. Towards Open World Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Amini-Naieni, N.; Amini-Naieni, K.; Han, T.; and Zisserman, A. 2023. Open-world Text-specified Object Counting. In *Proceedings of the British Machine Vision Conference*.

Amini-Naieni, N.; Han, T.; and Zisserman, A. 2024. CountGD: Multi-Modal Open-World Counting. In *Advances in Neural Information Processing Systems*.

Dai, S.; Liu, J.; and Cheung, N.-M. 2024. Referring Expression Counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Dave, A.; Khurana, T.; Tokmakov, P.; Schmid, C.; and Ramanan, D. 2020. TAO: A Large-Scale Benchmark for Tracking Any Object. In *Proceedings of the European Conference on Computer Vision*.

Deitke, M.; Clark, C.; Lee, S.; Tripathi, R.; Yang, Y.; Park, J. S.; Salehi, M.; Muennighoff, N.; Lo, K.; Soldaini, L.; Lu, J.; Anderson, T.; Bransom, E.; Ehsani, K.; Ngo, H.; Chen, Y.; Patel, A.; Yatskar, M.; Callison-Burch, C.; Head, A.; Hendrix, R.; Bastani, F.; VanderBilt, E.; Lambert, N.; Chou, Y.; Chheda, A.; Sparks, J.; Skjonsberg, S.; Schmitz, M.; Sarnat, A.; Bischoff, B.; Walsh, P.; Newell, C.; Wolters, P.; Gupta, T.; Zeng, K.-H.; Borchardt, J.; Groeneveld, D.; Nam, C.; Lebrecht, S.; Wittlif, C.; Schoenick, C.; Michel, O.; Krishna, R.; Weihs, L.; Smith, N. A.; Hajishirzi, H.; Girshick, R.; Farhadi, A.; and Kembhavi, A. 2025. Molmo and PixMo: Open Weights and Open Data for State-of-the-Art Vision-Language Models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Dendorfer, P.; Rezatofighi, H.; Milan, A.; Shi, J.; Reid, D. C. I.; Roth, S.; and Leal-Taixé, K. S. L. 2020. MOT20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*.

Djukic, N.; Lukezic, A.; Zavrtanik, V.; and Kristan, M. 2023. A Low-Shot Object Counting Network With Iterative Prototype Adaptation. In *Proceedings of the International Conference on Computer Vision*.

Dutta, A.; and Zisserman, A. 2019. The VIA Annotation Software for Images, Audio and Video. In *Proceedings of the ACM Multimedia Conference*, volume 27 of *MM 19*. ACM, New York, USA: ACM. To appear in Proceedings of the 27th ACM International Conference on Multimedia (MM 19).

Dwibedi, D.; Aytar, Y.; Tompson, J.; and Zisserman, A. 2024. OVR: A Dataset for Open Vocabulary Temporal Repetition Counting in Videos. *arxiv preprint arxiv:2407.17085*.

Dwibedi, D.; Aytar, Y.; Tompson, J. J. R.; Sermanet, P.; and Zisserman, A. 2020. Counting Out Time: Class Agnostic Video Periodicity in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Fang, Y.; Zhan, B.; Cai, W.; Gao, S.; and Hu, B. 2019. Locality-constrained Spatial Transformer Network for Video Crowd Counting. *arXiv preprint arXiv:1907.07911*.

Gong, S.; Zhang, S.; Yang, J.; Dai, D.; and Schiele, B. 2022. Class-Agnostic Object Counting Robust to Intraclass Diversity. In *Proceedings of the European Conference on Computer Vision*.

Han, T.; Bai, L.; Gao, J.; Wang, Q.; and Ouyang, W. 2022. DR.VIC: Decomposition and Reasoning for Video Individual Counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Heigold, G.; Keysers, D.; Minderer, M.; Lučić, M.; Gritsenko, A.; Yu, F.; Bewley, A.; and Kipf, T. 2023. Video OWL-ViT: Temporally-consistent open-world localization in video. In *Proceedings of the International Conference on Computer Vision*.

Hsieh, M.-R.; Lin, Y.-L.; and Hsu, W. H. 2017. Drone-Based Object Counting by Spatially Regularized Regional Proposal Network. In *Proceedings of the International Conference on Computer Vision*.

Jiang, R.; Liu, L.; and Chen, C. 2023. CLIP-Count: Towards Text-Guided Zero-Shot Object Counting. In *Proceedings of the ACM Multimedia Conference*.

Jones, L.; Elmore, J.; Boopalan, S. K.; Samiappan, S.; Evans, K.; Pfeiffer, M.; and Iglay, R. 2023. Controllable factors affecting accuracy and precision of human identification of animals from drone imagery. *Ecosphere*, 14.

Joseph, K. J.; Khan, S.; Khan, F. S.; and Balasubramanian, V. N. 2021. Towards Open World Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Kang, S.; Moon, W.; Kim, E.; and Heo, J.-P. 2024. VLCounter: Text-Aware Visual Representation for Zero-Shot Object Counting. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Kay, J.; Kulits, P.; Stathatos, S.; Deng, S.; Young, E.; Beery, S.; Van Horn, G.; and Perona, P. 2022. The Caltech Fish Counting Dataset: A Benchmark for Multiple-Object Tracking and Counting. In *Proceedings of the European Conference on Computer Vision*.

Levy, O.; and Wolf, L. 2015. Live Repetition Counting. In *Proceedings of the International Conference on Computer Vision*.

Li, J.; Yu, E.; Chen, S.; and Tao, W. 2025. OVTR: End-to-End Open-Vocabulary Multiple Object Tracking with Transformer. In *Proceedings of the International Conference on Learning Representations*.

Li, S.; Fischer, T.; Ke, L.; Ding, H.; Danelljan, M.; and Yu, F. 2023. OVTrack: Open-Vocabulary Multiple Object Tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Li, S.; Ke, L.; Danelljan, M.; Piccinelli, L.; Segu, M.; Van Gool, L.; and Yu, F. 2024a. Matching Anything By Segmenting Anything. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Li, S.; Ke, L.; Yang, Y.-H.; Piccinelli, L.; Segu, M.; Danelljan, M.; and van Gool, L. 2024b. SLAck: Semantic, Location, and Appearance Aware Open-Vocabulary Tracking. In *Proceedings of the European Conference on Computer Vision*.

Lin, W.; Yang, K.; Ma, X.; Gao, J.; Liu, L.; Liu, S.; Hou, J.; Yi, S.; and Chan, A. 2022. Scale-Prior Deformable Convolution for Exemplar-Guided Class-Agnostic Counting. In *Proceedings of the British Machine Vision Conference*.

Liotti, E.; Arteta, C.; Zisserman, A.; Lui, A.; Lempitsky, V.; and Grant, P. S. 2018. Crystal nucleation in metallic alloys using x-ray radiography and machine learning. *Science advances*, 4(4).

Liu, C.; Zhong, Y.; Zisserman, A.; and Xie, W. 2022. CounTR: Transformer-based Generalised Visual Counting. In *Proceedings of the British Machine Vision Conference*.

Liu, S.; Zeng, Z.; Ren, T.; Li, F.; Zhang, H.; Yang, J.; Li, C.; Yang, J.; Su, H.; Zhu, J.; et al. 2024a. Grounding DINO: Marrying dino with grounded pre-training for open-set object detection. In *Proceedings of the European Conference on Computer Vision*.

Liu, X.; Li, G.; Qi, Y.; Yan, Z.; Han, Z.; van den Hengel, A.; Yang, M.-H.; and Huang, Q. 2024b. Weakly Supervised Video Individual Counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Loy, C. C.; Chen, K.; Gong, S.; and Xiang, T. 2013. Crowd Counting and Profiling: Methodology and Evaluation. In *Modeling, Simulation and Visual Analysis of Crowds*.

Lu, E.; Xie, W.; and Zisserman, A. 2018. Class-agnostic Counting. In *Proceedings of the Asian Conference on Computer Vision*.

Makhura, O. J.; and Woods, J. C. 2019. Video Object Counting Dataset. In *IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*.

Meinhardt, T.; Kirillov, A.; Leal-Taixé, L.; and Feichtenhofer, C. 2022. TrackFormer: Multi-Object Tracking With Transformers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Minderer, M.; Gritsenko, A.; and Houlsby, N. 2023. Scaling Open-Vocabulary Object Detection. In *Advances in Neural Information Processing Systems*.

Minderer, M.; Gritsenko, A.; Stone, A.; Neumann, M.; Weissenborn, D.; Dosovitskiy, A.; Arnab, A.; Dehghani, M.; Shen, Z.; Wang, X.; Zhai, X.; Kipf, T.; and Houlsby, N. 2024. Simple Open-Vocabulary Object Detection with Vision Transformers. In *Proceedings of the European Conference on Computer Vision*.

Mustafa, O.; Braun, C.; Esefeld, J.; Knetsch, S.; Maercker, J.; Pfeifer, C.; and Rümmler, M.-C. 2019. Detecting Antarctic Seals and Flying Seabirds by UAV. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Nguyen, T.; Pham, C.; Nguyen, K.; and Hoai, M. 2022. Few-Shot Object Counting And Detection. In *Proceedings of the European Conference on Computer Vision*.

Pelhan, J.; Lukežič, A.; Zavrtanik, V.; and Kristan, M. 2024a. DAVE – A Detect-and-Verify Paradigm for Low-Shot Counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Pelhan, J.; Lukežič, A.; Zavrtanik, V.; and Kristan, M. 2024b. A Novel Unified Architecture for Low-Shot Counting by Detection and Segmentation. In *Advances in Neural Information Processing Systems*, volume 37. Curran Associates, Inc.

Qian, Z.; Han, R.; Hou, J.; Song, L.; and Feng, W. 2024. VOVTrack: Exploring the Potentiality in Videos for Open-Vocabulary Object Tracking. *arXiv preprint arXiv:2410.08529*.

Ranjan, V.; Sharma, U.; Nguyen, T.; and Hoai, M. 2021. Learning To Count Everything. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Ravi, N.; Gabeur, V.; Hu, Y.-T.; Hu, R.; Ryali, C.; Ma, T.; Khedr, H.; Rädle, R.; Rolland, C.; Gustafson, L.; Mintun, E.; Pan, J.; Alwala, K. V.; Carion, N.; Wu, C.-Y.; Girshick, R.; Dollár, P.; and Feichtenhofer, C. 2025. SAM 2: Segment Anything in Images and Videos. In *Proceedings of the International Conference on Learning Representations*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Schroeder, A. K.; Haugen, M. J.; Stettler, M. E. J.; and Boies, A. M. 2020. Using Computer Vision with Instantaneous Vehicle Emissions Modelling. In *2020 Forum on Integrated and Sustainable Transportation Systems (FISTS)*.

Schroeder, A. K.; Woodward, H.; Cornec, C. M. L.; Proust, T.; Benie, P. J.; Fan, S.; Aristodemou, E.; Jones, R. L.; Linden, P. F.; de Nazelle, A.; Boies, A. M.; and Stettler, M. E. 2024. Vehicle emission models alone are not sufficient to understand full impact of change in traffic signal timings. *Atmospheric Environment: X*.

Shi, M.; Lu, H.; Feng, C.; Liu, C.; and CAO, Z. 2022. Represent, Compare, and Learn: A Similarity-Aware Framework for Class-Agnostic Counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Sinha, S.; Stergiou, A.; and Damen, D. 2024. Every Shot Counts: Using Exemplars for Repetition Counting in Videos. *arXiv preprint arXiv 2403.18074*.

Sundararaman, R.; De Almeida Braga, C.; Marchand, E.; and Pettré, J. 2021. Tracking Pedestrian Heads in Dense Crowd. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Wen, L.; Du, D.; Zhu, P.; Hu, Q.; Wang, Q.; Bo, L.; and Lyu, S. 2021. Detection, Tracking, and Counting Meets Drones in Crowds: A Benchmark. In *CVPR*.

Wich, S. A.; Hudson, M.; Andrianandrasana, H.; and Longmore, S. N. 2021. Drones for Conservation. In *Conservation Technology*. Oxford University Press. ISBN 978-0-19-885024-3.

Yang, C.-Y.; Huang, H.-W.; Chai, W.; Jiang, Z.; and Hwang, J.-N. 2024. SAMURAI: Adapting Segment Anything Model for Zero-Shot Visual Tracking with Motion-Aware Memory. *arXiv preprint arXiv:2411.11922*.

Yang, S.-D.; Su, H.-T.; Hsu, W. H.; and Chen, W.-C. 2021. Class-agnostic Few-shot Object Counting. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*.

You, Z.; Yang, K.; Luo, W.; Lu, X.; Cui, L.; and Le, X. 2023. Few-Shot Object Counting With Similarity-Aware Feature Enhancement. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*.

Zhang, Y.; Shao, L.; and Snoek, C. G. 2021. Repetitive activity counting by sight and sound. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14070–14079.

Zhang, Y.; Sun, P.; Jiang, Y.; Yu, D.; Weng, F.; Yuan, Z.; Luo, P.; Liu, W.; and Wang, X. 2022. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. In *ECCV*.

Zhang, Y.; Zhou, D.; Chen, S.; Gao, S.; and Ma, Y. 2016. Single-Image Crowd Counting via Multi-Column Convolutional Neural Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Zhizhong, H.; Mingliang, D.; Yi, Z.; Junping, Z.; and Hongming, S. 2024. Point, Segment and Count: A Generalized Framework for Object Counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Zhou, X.; Girdhar, R.; Joulin, A.; Krähenbühl, P.; and Misra, I. 2022. Detecting Twenty-thousand Classes using Image-level Supervision. In *ECCV*.

Zhu, P.; Peng, T.; Du, D.; Yu, H.; Zhang, L.; and Hu, Q. 2021a. Graph Regularized Flow Attention Network for Video Animal Counting From Drones. In *IEEE Transactions on Image Processing*.

Zhu, P.; Wen, L.; Du, D.; Bian, X.; Fan, H.; Hu, Q.; and Ling, H. 2021b. Detection and tracking meet drones challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Appendix

A Additional Quantitative Results

Here we include additional quantitative results for which there was no room in the main paper.

A.1 ShanghaiTech Counting Results

In table 6, we evaluate the detection models Grounding DINO (Liu et al. 2024a) and OWLv2 (Minderer, Gritsenko, and Houlsby 2023), and the counting models CLIP-Count (Jiang, Liu, and Chen 2023) and CountGD-Box, on both Part A and Part B of the ShanghaiTech crowd counting dataset (Zhang et al. 2016). This table clearly shows that the counting models achieve lower counting errors than the detection models without training on any of the images in ShanghaiTech. For the CLIP-Count result, we use the published results from the CLIP-Count paper from training CLIP-Count on FSC-147 and testing it zero-shot on ShanghaiTech. For CountGD-Box and Grounding DINO, we use the text 'human'. For OWLv2, we use the text 'person', since it significantly improves its results compared to using 'human'.

Method	Prompt	ShanghaiTech Test			
		Part A		Part B	
		MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
GDINO	text	394.9	537.5	58.3	99.3
OWLv2	text	420.2	553.3	81.5	126.5
CLIP-Count	text	192.6	308.4	45.7	77.4
CGD-B	text	**132.2**	**253.9**	**32.2**	**57.9**

Table 6: Zero-shot Counting results for detection and counting models on the ShanhaiTech crowd counting dataset. 'GDINO' is Grounding DINO and 'CGD-B' is CountGD-Box.

A.2 FSCD-147 Validation Set Results

In table 7, we include the results from evaluating object detection and counting models on the validation set of FSCD-147 (Nguyen et al. 2022; Ranjan et al. 2021). For text only, CountGD-Box achieves the best performance on the validation set. For exemplars only, CountGD-Box is competitive with GeCo (Pelhan et al. 2024b). For both exemplars and text, CountGD-Box preserves the counting accuracy of CountGD (Amini-Naieni, Han, and Zisserman 2024) while significantly improving its detection accuracy. As stated in the main paper and as shown in table 6, the detectors perform poorly compared to the counters.

A.3 Temporal Window Length w Sensitivity

Here we test how the window length w of the temporal filter affects counting accuracy. For this experiment we use the first 10 videos from TAO-Count corresponding to 18 category-video pairs. Due to computational limitations, we do not run this test on the full TAO-Count dataset. The frame rate is held at 3 fps as it was set in the main paper while the window length for the temporal filter is varied from 1-6 frames. In this experiment, CountVid is implemented with CountGD-Box and SAM 2.1.

As shown in table 8, wider window lengths result in lower counting errors. This could be because in TAO-Count real objects tend to persist for at least two seconds and any frame-level detections that do not persist for this long are false positives. False positives that persist for more than one second (3 frames) but less than two seconds (6 frames) will only be removed by a temporal filter with a window length greater than 3. However, larger temporal filter windows result in lengthier inference times. Therefore, one needs to consider a balance between efficiency and accuracy when determining the length of the temporal filter.

A.4 Frame Rate Sensitivity

In table 9, we report the counting errors of CountVid, implemented using CountGD-Box and SAM 2.1, under different frame sampling rates on the Science-Count dataset. In this experiment, we provide text-only prompts to CountVid (i.e., no exem-

Method	Prompt	FSCD-147 Val			
		Counting		Detection	
		MAE ↓	RMSE ↓	AP ↑	AP50 ↑
GDINO	text	54.45	137.12	6.66	10.26
OWLv2	text	49.02	131.41	11.39	20.33
PSeCo	text	23.90	100.30	✗	✗
DAVE$_{prm}$	text	14.87	**59.60**	16.31	47.12
CountGD	text	12.46	67.52	13.36	43.90
CGD-B	text	**12.24**	66.24	**27.02**	**60.73**
C-DETR	exemp.	20.38	82.45	17.27	41.90
PSeCo	exemp.	15.31	58.34	32.12	60.02
DAVE	exemp.	9.75	**40.30**	24.20	61.08
GeCo	exemp.	9.52	43.00	**33.51**	62.51
CountGD	exemp.	9.34	51.74	15.16	47.96
CGD-B	exemp.	**9.27**	52.15	32.09	**68.82**
CountGD	both	**8.69**	**43.89**	14.74	47.95
CGD-B	both	8.76	44.24	**33.10**	**71.38**

Table 7: Results on FSCD-147 Val for image counting methods that output boxes (for Stage 1). ✗ indicates results are not provided by the authors. 'GDINO' is Grounding DINO, 'CGD-B' is COUNTGD-BOX, and 'C-DETR' is Counting-DETR (Nguyen et al. 2022). SAM masks are not used in evaluation. 'exemp.' means exemplars only, and 'both' means both exemplars and text as prompts.

Temporal Filter Window Length w	Prompt	TAO-Count *(10-video subset)*	
		MAE ↓	RMSE ↓
6 frames	text	4.4	10.3
3 frames	text	6.5	17.3
1 frame	text	14.2	34.4

Table 8: COUNTVID video counting results on the first 10 videos in TAO-Count using different temporal filter window lengths and given only text.

Frame Rate	Prompt	Penguins		Crystals	
		MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
6 fps	text	4.0	5.4	69.1	86.0
3 fps	text	4.0	5.3	65.9	84.1
1 fps	text	0.7	0.8	69.6	86.1

Table 9: COUNTVID video counting results on Science-Count using different frame rates and given only text.

plars are used). The original setup in the main paper uses a frame rate of 3 fps with a temporal window length w of 3 frames (i.e., 1 second) for Penguins, and 6 fps with a w of 3 frames (0.5 seconds) for Crystals. Here, we maintain a fixed *temporal duration* for the filter window—1 second for Penguins and 0.5 seconds for Crystals—across all tested frame rates.

For Penguins, we observe a significant improvement in counting accuracy as the frame rate decreases. Sampling at 6 fps leads to substantially higher errors than at 1 fps. A likely explanation is that higher frame rates introduce more noisy frame-level predictions (due to small motion or detection artifacts), which propagate and accumulate throughout the full video. Sampling fewer, more spaced-out frames suppresses this noise, making the aggregated prediction more robust.

For Crystals, the relationship between frame rate and counting accuracy is less straightforward. Reducing the frame rate from 6 fps to 3 fps improves performance, but further reducing it to 1 fps slightly increases the error. One explanation is that the 1 fps setting may cause the model to miss crystals that appear and disappear within a 1-second window, increasing counting error. Alternatively, lower temporal resolution could hurt tracking and association by skipping over intermediate frames.

Overall, these results indicate that the impact of frame rate is *dataset-specific*. Stable scenes like Penguins benefit from reduced frame rates, while rapidly changing scenes like Crystals require a balance between temporal coverage and noise suppression. Selecting the frame rate based on a dataset-specific validation set to maximize both accuracy and computational efficiency is a promising direction for future work.

B Further Analysis of Baseline Performance

B.1 MASA (Li et al. 2024a)

For MASA, we notice a bias towards predicting higher object counts than the ground truth on MOT20-Count. Grounding DINO (Liu et al. 2024a) produces unstable detections on the frames in MOT20-Count, since it is poor at detecting objects in crowded scenes. This results in lost tracks and the initialization of new tracks with new IDs, erroneously increasing the count. COUNTVID mitigates this issue by leveraging counting models that provide more persistent and stable detections in crowded scenes than detection models do. COUNTVID also leverages the more complex memory and occlusion prediction mechanisms of SAM 2.1 (Ravi et al. 2025) to retain and re-identify objects.

B.2 DR. VIC (Han et al. 2022)

DR. VIC is trained specifically for crowd counting and uses one of the MOT20-Count videos for validation and one for training, leaving only one-third of the videos in MOT20-Count as truly held out. Despite this, we test DR. VIC on all three videos in MOT20-Count, giving it an unfair advantage over COUNTVID, which is tested zero-shot on MOT20-Count. Then why does COUNTVID perform so much better than DR. VIC on MOT20-Count? We hypothesize this is because COUNTVID effectively leverages foundation models such as CountGD (Amini-Naieni, Han, and Zisserman 2024) (that uses the frozen image and text backbones of Grounding DINO, while finetuning the remaining modules on counting data) and SAM 2.1, which are trained on large quantities of data. On the other hand, DR. VIC is only trained and validated on the four training videos in HT21 (Sundararaman et al. 2021). This results in COUNTVID being a more accurate and general counting model that achieves superior performance on MOT20-Count than DR. VIC despite DR. VIC's unfair advantage.

C Further Implementation Details

C.1 COUNTVID Implementation Details

Here we detail the implementation details for each of the variants of COUNTVID used in table 5 of the main paper.

COUNTGD-BOX/SAM 2.1 (Main) Variant. For the main variant of COUNTVID used for comparing against baselines we set the hyperparameters as follows. For COUNTGD-BOX, we set the confidence threshold to 0.23, which is the same as the threshold for CountGD (Amini-Naieni, Han, and Zisserman 2024). We do not do any tuning of this threshold. For training, we start with the pretrained CountGD checkpoint published by the authors and use the same open-source training code, number of epochs (30) and seed. We set λ_{loc}, λ_{GIoU}, and λ_{cls} to 5, 2, 2 respectively. These settings were selected after trying the combinations $(\lambda_{loc} = 5, \lambda_{GIoU} = 2, \lambda_{cls} = 2)$, $(\lambda_{loc} = 1, \lambda_{GIoU} = 2, \lambda_{cls} = 5)$ and $(\lambda_{loc} = 1, \lambda_{GIoU} = 0, \lambda_{cls} = 5)$ and choosing the combination that achieved the lowest counting MAE on the FSC-147 (Ranjan et al. 2021) image counting validation set. We only train the model once and then use it for inference on VIDEOCOUNT without any fine-tuning on the videos in VIDEOCOUNT. For SAM 2.1 (Ravi et al. 2025), we use the $sam2.1\_hiera\_large.pt$ checkpoint and code published by the authors.

To speed up the experiments, we set the frame sampling rate for all the videos in VIDEOCOUNT to three frames per second, except the videos in the Crystals dataset. For those videos, the sampling rate is 6-7 frames per second, due to how rapidly the crystals form and disappear. The videos were provided to us pre-sampled at this frame rate. The window size w for the temporal filter is set to 3 for all the videos in VIDEOCOUNT, corresponding to one second for most of the videos. These hyperparameters were not tuned for optimal accuracy on VIDEOCOUNT. Future work could include developing a video counting validation set for this purpose.

COUNTGD-BOX/SAM 2 Variant. This variant is the same as the main variant, but uses the published $sam2\_hiera\_large.pt$ checkpoint instead of the $sam2.1\_hiera\_large.pt$ checkpoint published by the authors of SAM 2.

GeCo/SAM 2.1 Variant. This variant is the same as the main variant except that we use the GeCo (Pelhan et al. 2024b) counting and detection model. For GeCo, we use the pretrained checkpoint and code published by the authors. We use the same division threshold of 8 as the authors for almost all of the videos in VIDEOCOUNT except for Crystals due to too many false positives. For Crystals we set the division factor to 2.

GeCo/SAM 2 Variant. This variant is the same as the GeCo/SAM 2.1 variant except we use the published $sam2\_hiera\_large.pt$ checkpoint instead of the $sam2.1\_hiera\_large.pt$ checkpoint published by the authors of SAM 2.

COUNTGD-BOX/ByteTrack Variant. This variant is the same as the main variant except we replace the tracker with ByteTrack (Zhang et al. 2022). Since ByteTrack does not provide segmentation masks, we do not use the original new object detection logic and temporal filter from COUNTVID. Instead, we use ByteTrack's inbuilt tracking mechanisms. We still use COUNTGD-BOX to produce detections, which are then tracked with ByteTrack. The number of unique tracks identified in the video are enumerated to predict the final count. We use the implementation of ByteTrack provided by the Roboflow Supervision library.

C.2 Baseline Implementation Details

MASA (Li et al. 2024a). For the MASA baseline, we used the Grounding DINO (Liu et al. 2024a) checkpoint and code published by the authors with the Grounding DINO features and detections. Before deploying MASA on VIDEOCOUNT, we reproduced the results reported in the MASA repository for the pretrained checkpoint. We then adapted the code for the published demo to test MASA on VIDEOCOUNT. In our adapted code, the unique tracks output by MASA are enumerated to obtain the final count given the same text as all the methods compared to it. We set the $memo\_tracklet\_frames$ to 300 to help with retaining object identities for objects that disappear for long periods. All the other hyperparameters are set to the default values in the published demo. We use these hyperparameters for all the videos in VIDEOCOUNT.

DR. VIC (Han et al. 2022). For DR. VIC, we use the code and HT21 checkpoint published by the authors. Before deploying DR. VIC on MOT20-Count, we reproduce the published results on the HT21 test set. As noted above, the authors train on the MOT20-05 sequence in MOT20-Count (HT21-02 in HT21), in addition to the other training videos, and use the MOT20-01 sequence in MOT20-Count (HT21-01 in HT21) for validation. Here, we document and discuss our results using DR. VIC for each video of MOT20-Count for reference.

Sequence	Pred	GT	Absolute Error ↓	Relative Error ↓
MOT20-01^V	112.8	80	32.8	0.4
MOT20-02	418.1	278	140.4	0.5
MOT20-05^T	1329.7	1203	126.7	0.1

Table 10: DR. VIC counting results on the sequences in MOT20-Count. The V superscript on MOT20-01 indicates it was used for validation, and the T superscript on MOT20-05 indicates it is in the training set for DR. VIC.

As expected, in table 10, it shows that DR. VIC achieves the highest error on the held-out MOT20-02 sequence. In fact, this error is very close to the reported mean absolute error of 141.1 (table 2 of (Han et al. 2022)) of DR. VIC on HT21. The predicted count of 112.8 is also almost exactly the reported count of 113.0 on the equivalent HT21-01 sequence in the same table 2 of the DR. VIC paper. Finally, the authors include a qualitative result of DR. VIC's performance on a clip of MOT20-05 in the published repository, and the result for MOT20-05 in table 10 is consistent with this result.

C.3 New Object Detection

In fig. 4, we illustrate how COUNTVID is able to detect new objects. In more detail, in frame 0 fig. 4 (a) there are initially two fish. In frame M, a new blue fish appears. In fig. 4 (b) masks are predicted and verified for the fish independently for each frame in Stage 2, correctly segmenting the fish in each frame. In fig. 4 (c), the two masks for the fish in frame 0 are propagated in Stage 3 from frame 0 to frame M (indicated by the red arrows), and the masklets overlap with the independent object masks for the two fish. However, there is no mask propagated to overlap with the blue fish, since it was not in frame 0. Hence, the blue fish is considered a new object.

C.4 MAE & RMSE for Multi-Class Videos

While in the main paper, we define the video-based MAE and RMSE for the case where only one class is labeled per video,

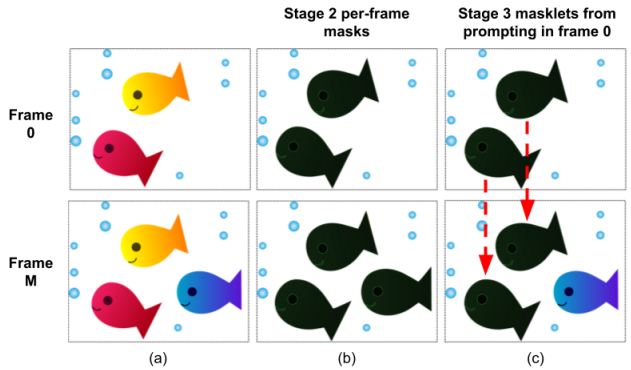

Figure 4: New object detection logic. A new blue fish that did not appear in frame 0 is detected in frame M, since no propagated mask from frame 0 overlaps with the blue fish in frame M (c).

here we explain how these metrics are calculated when multiple classes are labeled per video. Instead of averaging over all videos, in the multi-class case, we average over all category-video pairs. This means that if a single video has multiple objects, then the error from each category-video pair is treated equally. For example, if video one has cats and dogs labeled and video two has only fish labeled, then the total MAE and RMSE for these two videos would consist of three terms, each weighted equally. The first term would correspond to the error from video one for counting cats. The second term would come from video one for counting dogs. The third term would come from video two for counting fish. Each term would have a weight of one. Notice that this also means that because video one has more objects labeled, errors from it contribute more to the final MAE and RMSE than errors from video two.

D Computational Requirements

CountVid runs inference in three stages. Here we detail the time and memory requirements of CountVid for each stage. We run CountVid implemented with CountGD-Box and SAM 2.1 (large) on 1 NVIDIA RTX 3090 GPU with 24 Gb of GPU memory and 16 CPU cores. We use the Ubuntu 22.04.4 LTS operating system. We use the latest versions of the CountGD and SAM 2 codebases. The time for each stage given these resources for a 30-second video of 39 penguins is detailed in column 1 of table 11. The projected time for each stage given 8 GPUs and 128 CPU cores is given in column 2 of table 11. In general, the time and memory requirements of CountVid are governed by the requirements of the base counting and tracking models.

Stage 1. In Stage 1, the counting and segmentation models are applied to each frame independently. Therefore, operations can be completed in parallel over multiple batches and GPUs. For a fixed batch size, the time and memory for this stage scales linearly with the number of frames (video length) and the number of objects. In our setup, as shown in column 1 of table 11, Stage 1 takes 1 minute and 19 seconds and 17 GB of GPU memory for a 30-second video of 39 penguins. Given 8 GPUs, this time is divided by 8 in column 2.

Stage 2. In Stage 2, a temporal filter is applied to each detection in Stage 1 to remove false positives. We break up Stage 2 into Stage 2a, which uses GPU(s), and Stage 2b, which uses only CPU(s).

In Stage 2a, each detection in Stage 1 is tracked forward w frames and backward w frames from the original frame the detection is in. This results in processing $2w - 1$ unique frames for each detection. There is no redundant processing in that for each detection, we track it in a particular frame only once. Since each detection is tracked independently and frames are processed independently, operations can be completed in parallel over multiple batches of frames and objects over multiple GPUs. For a fixed batch size, the time for this stage scales linearly with the number of frames, the number of objects, and the filter width w. The memory scales linearly with the number of objects. In our setup, as shown in column 1 of table 11, Stage 2a takes 11 minutes and 19 seconds and 17 GB of GPU memory for a 30-second video of 39 penguins. Given 8 GPUs, this time is divided by 8 in column 2.

In Stage 2b, each propagated mask from Stage 2a is matched with per-frame masks from Stage 1. Since masks are checked independently, they can be processed in parallel. This stage requires no GPUs. The time for this stage scales linearly with the number of frames and the number of objects. In our setup, as shown in column 1 of table 11, Stage 2b takes 7 minutes and 31 seconds for a 30-second video of 39 penguins. Given 128 CPUs, this time is divided by 8 in column 2.

We also consider the inference time for CountVid when Stage 2 is removed in the last row of table 11. In applications where processing time is more important than removing false positives, CountVid can be applied without Stage 2, resulting in faster execution times.

Stage 3. In Stage 3, the video is processed iteratively, so there is limited opportunity for parallelism. Furthermore, the degree of parallelization possible depends on the tracking model. SAM 2 and SAM 2.1 treat each object independently. This means that masks from one object cannot influence the masks from another object. As a result, the objects can be propagated in parallel, in separate batches, and on separate GPUs.

In a non-causal setup, all the objects detected in the first frame can be propagated in parallel for the whole video. Then, for each frame after the first frame, CountVid checks for new objects, propagates them in parallel, and combines them with the masks of the other objects. This continues iteratively throughout the video. Since the frames are processed iteratively, and each object is propagated once, the time and memory for this phase scale linearly with the number of frames and the number of objects.

In our setup, as shown in column 1 of table 11, Stage 3 takes 6 minutes and 37 seconds and 17 GB of GPU memory for a 30-second video of 39 penguins. To project the time required given 8 GPUs, we divide time required to propagate the objects in the initial frame by 8 and add that to the time required to process the rest of the video in our setup with 1 GPU.

E Further Dataset Details

Here we include further details on VideoCount. In fig. 5, we include sample frames from VideoCount. In table 12, we compare VideoCount to other counting datasets. In fig. 6, we show VideoCount spans a wide range of categories and counts. While VideoCount includes more videos with fewer than ten objects than those with more, this imbalance reflects a common challenge in object counting datasets. For example, FSC-147 (Ranjan et al. 2021), a well-established image counting dataset, also exhibits significant imbalance due to the inherent difficulty of annotating high object counts. This challenge is further amplified in videos, where annotating counts across hundreds of frames involves considerable effort. Despite this, we include videos with annotations for up to 1203 objects, a large number for video counting benchmarks. To avoid any bias toward sparse scenes, we report results separately

Table 11: Inference time for running CountVid on a 30-second video of 39 penguins. Stage 2a refers to tracking forwards and backwards in time for each detection in Stage 1. Stage 2b refers to matching the propagated masks from Stage 2a with the per-frame masks from Stage 1 to identify and remove false positives. All projected timings in the rightmost column are estimated analytically; only the 1-GPU, 16-CPU results are implemented and measured directly.

	1 GPU, 16 CPU Cores	8 GPUs, 128 CPU Cores (Projected Analytical Estimate)
Stage 1	1m 19s	10s
Stage 2a	11m 19s	1m 25s
Stage 2b	7m 31s	56s
Stage 3	6m 37s	2m 27s
Total + Stage 2	26m 46s	4m 58s
Total - Stage 2	7m 56s	2m 37s

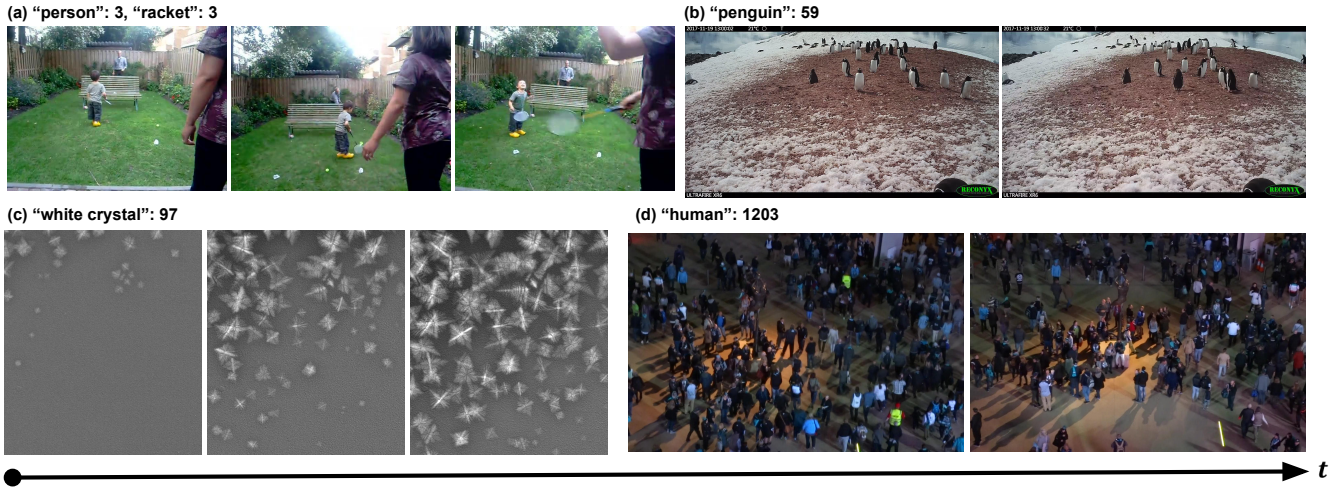

Figure 5: Examples from VideoCount. (a) TAO-Count includes multiple moving objects (rackets and players). Science-Count contains videos of (b) similar-looking penguins of varying sizes and (c) crystals in x-ray videos that change structure over time. (d) MOT20-Count involves counting over a thousand people in dense crowds. VideoCount poses diverse, high-density, and dynamic challenges for open-world counting.

Dataset	Video?	# of Samples	# of Classes	Min. # Objects per Sample	Max. # of Objects per Sample
FSC-147	✗	6135	147	7	3731
CARPK	✗	1448	1	1	188
VisDrone	✓	263	10	-	-
Mall	✓	1	1	60,000	60,000
VideoCount	✓	370	141	1	1203

Table 12: Comparing VideoCount to the other counting datasets: FSC-147, CARPK (Hsieh, Lin, and Hsu 2017), VisDrone (Zhu et al. 2021b), and Mall (Loy et al. 2013). A ‘sample’ refers to a single image if considering an image counting dataset or a single video if considering a video counting dataset. ‘-’ indicates that information is not accessible. Object counting datasets with large numbers of categories, as is expected in the open-world setting, are not available for our new task. While FSC-147 (Ranjan et al. 2021) covers 147 categories, it does not support videos. While VisDrone (Zhu et al. 2021b) and Mall (Loy et al. 2013) support videos, they are limited to 1-10 categories. Our proposed VideoCount overcomes this limitation by covering a large number of categories while also supporting videos.

for TAO-Count (1-10 objects) and MOT20-Count (80-1203 objects).

E.1 TAO-Count

TAO (Dave et al. 2020) contains 2,907 videos with tracks labeled for 833 different object categories. Due to its diversity of object types, we build TAO-Count on top of TAO to test how well open-world object counting methods generalize to many different categories. However, TAO has certain limitations. Firstly, it has on average 5.9 tracks per video, and the maximum number of tracks per video in our filtered subset of TAO is 10, which is considered very low for counting. For example, FSC-147 (Ranjan et al. 2021), the

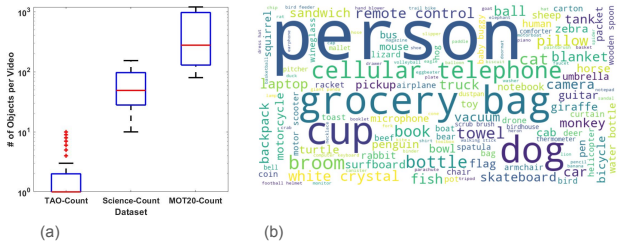

Figure 6: In (a), box plots show that the VIDEOCOUNT covers a large range of counts from 1-1203 objects per video. In (b), a word cloud shows the diversity of object classes covered. The 'person' class is quite common because it is one of the most popular exhaustively annotated classes in TAO-Count.

standard open-world object counting dataset for images, has on average 56 objects and at most 3,731 objects per image. Additionally, not all objects in TAO are exhaustively annotated, which is necessary for counting. To address the former issue, we also propose MOT20-Count. To solve the latter issue, we filter out categories and videos that are not exhaustively annotated from the TAO validation set. This process results in 357 videos covering 139 categories in TAO-Count. We do not include counts for the AVA and HACS videos as these videos require third-party approvals to obtain. To obtain the text input for the categories, we use the category synsets.

E.2 MOT20-Count

MOT20 (Dendorfer et al. 2020) contains 8 videos with heavily crowded scenes ranging from 17 seconds to a little over 2 minutes. To build MOT20-Count, we use three of the training videos, each with 80 to 1203 humans for evaluating counting methods on crowded scenes. The complete tracking annotations for MOT20 Test, which are required to get the final counts, are not publicly available as they are for MOT20 Train. This restriction is in place to prevent test set leakage and preserve the integrity of MOT20 as a well-established tracking benchmark. Despite this limitation, we chose to evaluate on a subset of the MOT20 training videos because they include dense scenes with many occlusions and overlapping pedestrians, critical for evaluating object counting in challenging conditions. To maintain fairness, we do not train any of the components of COUNTVID on MOT20.

The original MOT20 ground truth annotations include the 'crowd' class, where regions of the video considered too densely populated are grouped together into a single bounding box. Furthermore, static people, people in vehicles, and children in strollers are omitted from the MOT20 evaluation procedure, even though their ground truth boxes are provided. For the counting task, given the text 'human,' the model should count *all* the humans. Therefore, in MOT20-Count, we omit videos from MOT20 that include instances of the 'crowd' class. For the remaining videos, we add the numbers of unique trajectories for the 'pedestrian' class, 'static person class,' and 'person on vehicle' class to the count. We examine the bounding boxes for the remaining classes and add any humans that do not fall into the previous categories (e.g., children in strollers and disabled people in electric wheelchairs) to the count. We do not include children in strollers that are not visible throughout the entire video. The result of this process is what we consider to be the ground truth count.

Another note is that the original MOT20 annotations do not take into account re-identification–if humans leave and reenter the scene, they are relabeled as new instances. However, the maintainers of MOT20 have ensured us that such cases should happen very rarely if at all in the dataset. Furthermore, we incorporate re-identification challenges into TAO-Count. For example, there are videos where humans leave for more than 2 seconds and return to the scene with the same identity label in TAO-Count. Therefore, VIDEOCOUNT does explicitly challenge counting methods with re-identification tasks.

Finally, we compare our MOT20-Count to an existing crowd counting dataset that also has a comparably high average human count (> 500 per video) to show why it was necessary to build our own dataset. The tracking dataset CroHD (also known as HT21) (Sundararaman et al. 2021) has been used as a Video Individual Counting (VIC) dataset. However, the ground truth counts used from this dataset by VIC methods like DR. VIC (Han et al. 2022) have issues. Looking at table 2 of (Han et al. 2022), it is clear that the *total number of tracks* rather than the *total number of humans* is being used as the ground truth count. These tracks include tracks for statues and human faces on clothing. On the other hand, in MOT20-Count, we only count the number of living humans. At the same time, we leverage existing annotations from MOT20, to avoid reinventing existing datasets. Furthermore, these datasets do not include significant re-identification challenges, which is one of the reasons that it was necessary to add TAO-Count to our suite of benchmarks as it contains significant re-identification challenges. Further comparisons of our full VIDEOCOUNT to other benchmarks are presented in table 12.

E.3 Science-Count

Penguins. To test COUNTVID on real-world biodiversity applications, we include three 10-30 second videos of penguins in their natural habitats. These videos were initially taken to monitor the penguin population in a particular region over time. We sample the videos at 3 frames per second and annotate each frame using the VGG Image Annotator (Dutta and Zisserman 2019). Specifically, we add a single point inside each penguin for every frame. Penguins are then matched manually across frames. The total number of unique penguins identified in each video is the ground truth count for that video. Each video has 28-59 penguins.

Crystals. In Science-Count, we include new videos from the very challenging real-world setting of counting crystals that form from liquid metal alloys. Because this process is very rapid, we do not downsample the frames for annotation. Instead, we annotate the frames at 6-7 frames per second. The bounding boxes of the x-ray images are already provided. However, as the crystals form, many of them disappear while new ones enter. To account for this, we calculate the ground truth count by adding the number of crystals that disappear to the number of crystals in the final frame. In this setting, no crystals re-enter the scene, as they all move in the same direction out of the camera's field of view and never return. We provide data for seven videos each with 10-154 crystals.

F Clarifications

F.1 Open-World Repetition Counting in Videos

While open-world *repetition* counting in videos is an interesting problem with existing datasets like OVR (Dwibedi et al. 2024) for testing methods on it, our work is distinct from this task. Methods for *repetition* counting (Dwibedi et al. 2020; Levy and Wolf 2015; Sinha, Stergiou, and Damen 2024; Zhang, Shao, and Snoek 2021), as opposed to *object* counting, count the number of times a recurring action or motion event (e.g., a single jumping jack, clap of a hand, bounce of a ball) occurs in a video. Recent work (Sinha, Stergiou, and Damen 2024) has followed the evolution of models for

counting objects in images, by developing models where an exemplar clip can specify the type of repetition to count. However, unlike objects, events are defined by temporal, rather than spatial, boundaries, and the essence of the repetition counting task is *temporal self-similarity*. Consequently, the repetition counting task has more in common with object counting in images (which is concerned with spatial self-similarities) than with object counting in videos.

F.2 "Open-World" vs "Open-Vocabulary"

Here we make an important note about terminology. We notice that in the object counting literature (Amini-Naieni et al. 2023; Amini-Naieni, Han, and Zisserman 2024; Liu et al. 2022), *open-world counting* refers to the task of counting instances of an object class specified at test time via textual or visual prompts. These counting models are *open-world* because they generalize beyond a fixed vocabulary, as the object of interest may belong to a class not seen during training. However, crucially, the category is still explicitly provided as input at inference time, either as text or exemplar. We adopt this definition of *open-world* in our work, since CountVid builds on counting literature.

However, this usage of *open-world* differs from that in some earlier literature. The origins of the formal definition of 'open world' are from (Abhijit and Terrance 2015). This paper defines an open-world recognition system as one that recognizes instances of both *known* and *unknown* classes, marks the unknown objects as 'unknown,' obtains class labels for these unknown objects, and incrementally learns from these instances such that they become 'known.' Similarly, early works in open-world object detection (Joseph et al. 2021) aim to discover and learn novel object categories without labels at first, marking them as 'unknown,' and incrementally learning them later. However, more recent works in detection (Minderer, Gritsenko, and Houlsby 2023; Minderer et al. 2024) use the term 'open-world' to describe a detection model that can detect objects unseen during training by accepting textual prompts (i.e., labels) describing the object. Importantly, this means that the labels of the unknown objects must be provided for the model to detect them, which differs from the original formal definition of 'open world' in (Abhijit and Terrance 2015). The OWL paper (Minderer et al. 2024) uses the terms 'open-world' and 'open-vocabulary interchangeably to describe this prompt-based setting as we do. In fact 'OWL' stands for 'Open-World Localization.' Like OWL, in our case, the category is always specified by the user, and the challenge lies in handling domain shifts, visual diversity, and lack of training-time exposure to the category.

G Discussion and Possible Extensions

G.1 Boxes vs. Points as Unambiguous Prompts

In general, bounding boxes are less ambiguous specifications of the object to count than points are. In fact, the whole field of visual-exemplar-based counting relies on this idea (Gong et al. 2022; Liu et al. 2022; Lu, Xie, and Zisserman 2018; Nguyen et al. 2022; Ranjan et al. 2021; Shi et al. 2022; Yang et al. 2021; You et al. 2023; Djukic et al. 2023; Lin et al. 2022). For these methods, it is assumed that a 'visual exemplar', bounding box over one example instance, is an unambiguous specification of the object to count. However, it is interesting to note that in some (very rare) cases, overlapping objects may share the same tight bounding box. For example, if one person stands directly behind another person with little to no part of their body showing, a bounding box could refer to either (a) the person in the front, (b) the person in the back, or (c) both people simultaneously (e.g., counting *pairs* of people rather than each person, one at a time). However, usually the IoU between bounding boxes of different objects will be less than one, allowing CountVid to unambiguously identify each object separately. This is much better than points, where the object can be identified as its whole or subparts. Future work could consider using an object *segmentation* to specify the prompt.

G.2 Causality

Here we discuss the aspect of causality and what would be required to make CountVid causal. By 'causal' we mean that CountVid provides the global count – the number of unique objects in the video – at any point in time given only the past frames. For CountVid to be causal, the base tracking model needs to be causal. For example, SAM 2 and SAM 2.1 process frames in a streaming fashion, and are, thus, causal. Given this, there are two approaches to allow for causality.

In the first approach, the global count is provided with a lag of w frames. Stage 1 of CountVid is applied to frames $1 \cdots t$, and Stages 2 and 3 are applied to frames $1 \cdots t - w$, producing a global count at $t - w$. The w-frame lag is to allow for filtering false positives in Stage 2. In practice, this lag corresponds to about 1 second.

In the second approach, the global count is provided immediately. Only Stages 1 and 3 of CountVid are applied to frames $1 \cdots t$, and Stage 2 is skipped. This represents a tradeoff between causality and accuracy as although the global count will be provided more quickly, more false positives will be included in it.

G.3 Object Quotas of Counters & Detectors

Here we briefly discuss how to overcome object quotas of counting and detection models. Detection models based on finite numbers of region proposals, such as Grounding DINO (Liu et al. 2024a) and OWLv2 (Minderer, Gritsenko, and Houlsby 2023), have quotas on the number of objects they can detect in a single frame. This is also true for counting models based on these detectors (Amini-Naieni, Han, and Zisserman 2024; Dai, Liu, and Cheung 2024). To overcome these limits, prior methods (Amini-Naieni, Han, and Zisserman 2024) propose cropping the image into smaller pieces and summing the counts for each crop while averaging the counts along the overlapping crop boundaries. The issue with these approaches is that averaging overlapping boundaries does not provide bounding boxes for these regions.

There are ways to address this. For example, predicted boxes with IoUs greater than 0.5 from different crops in overlapping regions can be assigned to one object with a bounding box formed by either averaging the coordinates of the boxes from the different crops, or choosing the box with the highest confidence score. Methods like this provide boxes for objects on the overlapping regions. Additionally, the crop size can be determined by first estimating the object size by averaging the sizes of the predicted bounding boxes and then choosing $crop\_size \ll obj\_size \times quota$. This approximately ensures that the number of objects in each crop is below the model's quota. Future work will extend counting and detection models to count numbers of objects above their quotas using such techniques and plugging these models into the CountVid framework.

G.4 Limitations

Because CountVid builds on a segmentation and tracking model (Ravi et al. 2025; Yang et al. 2024), it also suffers from some of its shortcomings. For example, the objects in SAM 2 are not 'aware' of each other. This means that SAM 2 cannot predict the mask for one object by leveraging information from the predictions for other objects. As a result, object masks can overlap with each other, even if each object has a distinct boundary. Furthermore, the efficiency of the counting and tracking models govern the

efficiency of COUNTVID. Batch processing objects and frames can improve the speed and memory consumption of the model. Since COUNTVID is a general framework, faster and smaller counters and trackers can be swapped in.

Another limitation is that while the video-based MAE and RMSE metrics proposed incorporate re-identification and matching, they do not incorporate identity switching. This is because accounting for identity swaps is not significant for calculating an accurate count. If two objects swap identities, the ground truth count does not change. While the scope of this paper focuses on counting, future work could investigate how well COUNTVID performs as a tracker by evaluating its identity consistency and trajectory accuracy.

Finally, COUNTVID performs detection at the frame level in Stage 1, and therefore this stage does not use temporal cues. This becomes a limitation in scenarios where objects are not visually distinguishable in individual frames but are only discernible through their motion over time. For example, in the Caltech Fish Counting Dataset (Kay et al. 2022), many fish are nearly indistinguishable from the background due to camouflage and low contrast. They become detectable only by observing their movement across sequential frames. As shown in fig. 7, COUNTGD-BOX fails to localize the fish in Stage 1 when provided only a single frame and the text "fish." Therefore, COUNTVID is unable to count the fish in later stages.

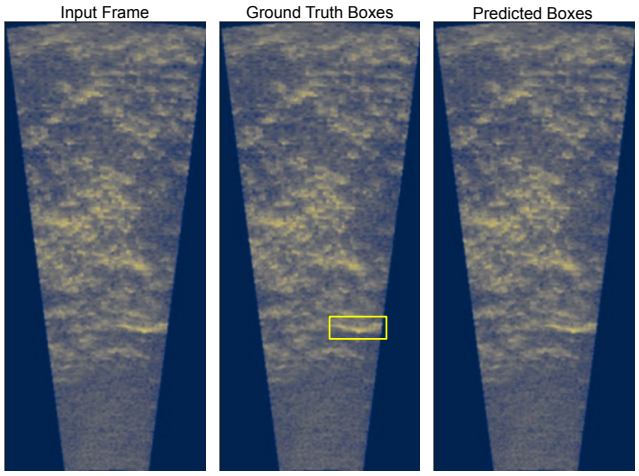

Figure 7: COUNTGD-BOX struggles to identify the fish in the video frame given the text "fish," since the fish is only clearly visible and distinct from the background when analyzing motion patterns in the water across the video.