# Effects of Variable Snapshot Frequency on Object Tracking

HEIN HUIJSKES, University of Twente, The Netherlands

## ABSTRACT

Various areas within artificial intelligence are increasingly adopting environmentally sustainable practices. Part of this movement involves reducing the energy footprint of algorithms, particularly in applications like object tracking. Current practices in evaluating object tracking typically aim for the maximum possible frame rate, which consumes a considerable amount of energy. A lower frame rate that is still effective can reduce environmental costs by lowering energy consumption from object detection algorithms, while still achieving an effective result. This research aims to introduce varied frame rates for camera-feed-based object detection algorithms. State-of-the-art object detection and object tracking algorithms are discussed to find a suitable algorithm for variable frame rate. Together with an in-the-field company a prototype for assessing variable frame rate-based object detection is built and tested using the discussed algorithms. This prototype shows at what lower frame rates object tracking could still be used to adequately detect objects. Different metrics measuring the accuracy of object tracking are compared at different frame rates. This has the potential to minimize hardware and software needs, and electricity consumption.

Additional Key Words and Phrases: Object Detection, Object Tracking, MOTChallenge

## 1 INTRODUCTION

While many studies have developed different benchmarks to assess the effectiveness of object tracking, they generally consider the maximum performance to run algorithms on (e.g., [14], [31], [32], and [36]). For object tracking specifically, this means running algorithms at the maximum possible frames per second (FPS). In efforts to increase sustainability in AI, some research has already been done to look into different factors that contribute to the environmental cost and computational consumption of AI (e.g., [21], [24], and [29]). Factors such as hardware and software limitations are considered alongside their performance and environmental impact. Still, while some variables describing the tracking models are considered, few studies go over lower frame rate specifically. The studies that do, tend to focus on higher frame rate [11], specific low-FPS implementations [30], specific industry-based implementations (e.g. frame interpolation for video-based vehicle counting [27]) or object detection metrics rather than object tracking [23]. Something that still seems to be missing here is a general overview of the impact of lower frame rates on object detection. That is where this research comes in, as it attempts to show how ranges of lower frame rates affect the accuracy of object detection.

Studying the impact of lower frame rates is relevant for assessing object tracking in the context of lowering the usage of computational resources. A 2019 study for example mentions that many of the best object tracking algorithms could not run in real-time, since their computation time per frame is higher than the FPS of the video input they receive [21]. This frame rate mismatch can be a challenge in industries that use object tracking in live feeds, such as construction site monitoring, where object tracking can be valuable [2]. In these industries, hardware can be limited, which subsequently makes computational power limited. It could therefore be valuable to assess if there is a middle ground with less computation that still generates results that are adequate enough to function in the respective industries. One of the ways to do this is simply lowering the frame rate of the video feed fed to the algorithms, and verify what the quality is of the results obtained from them. This research therefore addresses the following question:

- **What is the influence of varying frame rates on the accuracy of object tracking algorithms?**

This question is further refined to:

- Which state-of-the-art object detection and object tracking algorithms exist?
- What are adequate benchmarks to assess the accuracy of an object tracking algorithm?
- How can the accuracy and frame rate of object tracking algorithms be compared?

*TScIT 37, July 5, 2024, Enschede, The Netherlands*

## 2 BACKGROUND

### 2.1 Definitions

Object detection can be defined as a "computer vision task that deals with detecting instances of visual objects of a certain class [...] in digital images" [37]. An object detection model receives an image and returns the location and class of any detected objects in the image. Object tracking, also referred to as Multiple object tracking (MOT), adds one layer on top of this by attaching a unique ID to each object, where "the task of MOT is largely partitioned into locating multiple objects, maintaining their identities, and yielding their individual trajectories given an input video" [20]. This allows for tracking unique objects across multiple images. Although these methods can be intertwined [12], this study considers them as two separate steps to focus more on the effects of frame rate on object tracking. This means that object detection and object tracking are handled by separate algorithms, where the detection algorithm passes the information about detected objects to the tracking algorithm.

### 2.2 Object Detection

In the past years, various research has been conducted in the field of object detection and object tracking. Due to this, many improvements have been made in different applications of the algorithms both used individually and together [5], [37].

A 2023 survey of 20 years of object detection lines out the main developments in object detection [37]. Since 2014, most models have been using deep learning to detect objects in images. There are two main types of algorithms; two-stage detectors and one-stage detectors [37]. Two-stage detectors process images in multiple steps, where each step improves the knowledge the model has of the image. Such steps could for instance be first proposing possible locations for objects in a frame, and then in a second step verifying and refining these object locations. One-stage detectors instead aim to complete the detection process in one step. While two-stage detectors can achieve high precision easily, they tend to be slow and complex. This leads to most in-the-field solutions using one-stage detectors, which are generally faster but suffer when detecting small and dense objects. Six such one-stage detectors are You Only Look Once (YOLO) [25], Single Shot Multibox Detector (SSD) [17], RetinaNet [15], CornerNet [13], CenterNet [35], and DETR [4]. The speed and accuracy of all of these models are listed in Table 1. This table shows each model's highest reported FPS performance, and the corresponding accuracy in mean average precision (mAP). Each of the mentioned models was released in the respective listed order, and each reported an improved accuracy over its predecessors.

YOLO was one of the first one-stage detectors, introduced in 2015 [25]. It is fast compared to other detectors, but suffers in localization accuracy, especially on small objects. It has continuously received improvements, its latest official version (YOLOv8) was released in 2023 and is still being updated. This version is faster and more accurate compared to previous iterations [28]. SSD was introduced later in 2015, and aimed to simplify object detection by proposing default boxes for object detection. This meant it was fast, and could be easier to integrate into other systems [17]. RetinaNet is an algorithm developed after researching the reason for the accuracy difference between one-stage and two-stage detectors [15]. CornerNet introduced a new approach to object detection by detecting a bounding box as a pair of keypoints [13]. CenterNet builds on the idea of CornerNet, but treats objects as a single point instead, at the center of its bounding box [35]. DETR attempted to streamline the detection process by using bipartite matching for predictions, and using a encoder-decoder architecture [4].

Each model has its own advantages and particular uses. Relevant for this research is a sufficient accuracy combined with high FPS. The high FPS is relevant so that FPS is not the limiting factor for this model to run in real-time. The reported FPS seen in Table 3 is also generally self-reported by each of the models' authors, and usually run on high-end hardware. Since this research aims to provide results that may be useful to low-end hardware solutions too, a higher base FPS is a necessary consideration.

YOLO seems to have a high initial FPS and mAP, but was to the knowledge of the author only reportedly tested on the VOC [7] dataset [25]. The VOC dataset can be considered as easier to detect objects on than the COCO [16] dataset, since the COCO dataset generally has smaller objects [9]. Therefore YOLOv2 is also listed. YOLOv8 is also shown, since that is the latest official release of YOLO. It has higher FPS than other models and comparable mAP.

| Algorithm | FPS | mAP | Open Source | Dataset |
|---|---|---|---|---|
| YOLO | 155 | 52.7% | Yes | VOC |
| YOLOv2 | 40 | 21.6% | Yes | COCO |
| YOLOv8x | 280 | 53.9% | Yes | COCO |
| SSD | 59 | 46.5% | Yes | COCO |
| RetinaNet | 14 | 32.5% | Yes | COCO |
| CornerNet | 4 | 42.2% | Yes | COCO |
| CenterNet | 142 | 28.1% | Yes | COCO |
| DETR | 28 | 42% | Yes | COCO |

Table 1. Object detection algorithms comparison

| Algorithm | FPS | MOTA | Open Source |
|---|---|---|---|
| ByteTrack | 13.7 | 77.8 | Yes |
| BoT-SORT | 6.6 | 80.5 | Yes |

Table 2. Object tracking algorithms comparison

## 2.3 Object tracking

Object tracking or MOT, also referred to as Multi-Target Tracking (MTT), tracks the identities of objects across different frames using object detection. A 2020 survey explains the main differences between MOT algorithms [5]. Most algorithms focus only on identifying objects, leaving the actual object detection to state-of-the-art detectors such as YOLO, Faster-RCNN, and SSD. MOT algorithms can also be divided into batch and online methods. Batch algorithms are allowed to use future frames, meaning they have access to frames that come after the currently considered frame. This way they can use future frames to improve the accuracy of tracking in the current frame. Online methods, meanwhile, can only use current and past frames. This can mean lower accuracy for object tracking, focusing instead on being able to broadcast the results live. Usually, online methods are faster than batch methods since they require fewer computation steps. Still, both types are often too slow to run in real-time [11]. Two object tracking algorithms are ByteTrack [34] and BoT-SORT [1]. Both can be seen in Table 2, where there speed and accuracy are listed. Both algorithms operate using the online method, meaning that they run in real-time, tracking objects in frames solely based on past frames. The reported results in Table 2 also use publicly available datasets.
ByteTrack is a state-of-the-art object tracking model released in 2021 that performs well in terms of accuracy and speed.
BoT-SORT is an object tracking model that ranked highest on MOT leaderboards when it was released in 2022. It improves upon the scores of ByteTrack, but reportedly performs slower, as seen in Table 2.

## 2.4 Benchmarking

*2.4.1 General benchmarks.* Different benchmarks have been adapted and established for both object detection and object tracking. The benchmarks for detection are less relevant to this research since the frames fed to object detection remain the same over the different experimental configurations. This means that the accuracy of the detections remains the same for each setup, making it less relevant for this research other than a required baseline of accurate detections. The baseline is necessary to ensure that the results indicate effects of object tracking, and are not due to underlying issues of object detection. This can be measured using MOTP, an object tracking benchmark. Therefore only the benchmarks for tracking are covered here.

According to [5], for MOT "a group of metrics have been de facto established as standard, and they are used in almost every work". These metrics are the CLEAR MOT metrics [3] and ID metrics [26],

both of which are largely defined in terms of metrics introduced by Wu and Nevatia [33]. CLEAR MOT consists of Multiple object tracking Accuracy (MOTA) and Multiple object tracking Precision (MOTP). The ID metrics are Identification Precision (IDP), Identification Recall (IDR), and Identification F1 (IDF1).
To compute CLEAR MOT metrics, the following parameters are required [5]:

- FP: the number of false positives in the whole video
- FN: the number of false negatives in the whole video
- Fragm: the total number of fragmentations
- IDSW: the total number of ID switches

The CLEAR MOT metrics are calculated as follows [5]:

$$MOTA = 1 - \frac{FN + FP + IDSW}{GroundTruth}$$

$$MOTP = \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t}$$

Where $c_t$ is the number of matches in frame $t$, and $d_{ti}$ is the bounding box overlap between hypothesis $i$ with its assigned ground truth object [5]. To compute ID metrics, the following parameters are additionally required [5] (IDTN is technically not used, but included for the sake of completeness):

- IDTP: true positive ID matches
- IDFP: false positive ID matches
- IDFN: false negative ID matches
- IDTN: true negative ID matches

The three ID metrics are calculated as follows [5]:

$$IDP = \frac{IDTP}{IDTP + IDFP}$$

$$IDR = \frac{IDTP}{IDTP + IDFN}$$

$$IDF1 = \frac{2IDTP}{2IDTP + IDFP + IDFN}$$

The MOTChallenge framework (see 2.4.2) [14] lists additional metrics, as well as the components of the above metrics. These metrics are for instance Mostly Tracked targets (MT), Mostly Lost targets (ML), and Higher Order Tracking Accuracy (HOTA). These are not further listed since their insights are deemed to be less relevant (MT and ML), or their results are largely covered in other metrics (MOTA versus HOTA [19]). FPS is also listed at MOTChallenge as Hertz (Hz), however, this metric is more inconsistent. This may be due to differences in hardware and environments that the frameworks are run on, and choices of exclusion or inclusion of computation time of used detection algorithms.

*2.4.2 MOTChallenge.* In 2015, the first MOTChallenge, MOT15, was launched to publicly compare the performance of different object tracking algorithms [14]. It provides a common dataset containing videos of pedestrians, which algorithms could run object detection on. On their website (motchallenge.net) they list the rankings of each algorithm based on CLEAR MOT and other metrics. In subsequent years, MOT16, MOT17, and MOT20 would come out to offer improved and/or more difficult datasets to test tracking on [22] [6]. In addition, other more specific datasets and corresponding leaderboards were released specifically made for different areas of object tracking. MOTChallenge and its listed metrics are often cited to compare new algorithms that are published to older ones [5], [20]. This allows comparison of different object tracking algorithms based on their performance, which can aid in selecting algorithms to use for application or further research. It also provides a baseline for performance, allowing new research based on old algorithms to compare results against this baseline.

| Dataset | Images | FPS | Objects | Type |
|---|---|---|---|---|
| MOT15 | 5500 | 7-30 | 39905 | Pedestrian |
| MOT20 | 8931 | 25 | 1336920 | Pedestrian |
| KITTI | 7481 | 10 | 80265 | Traffic |
| PETS2009 | | 7 | | Pedestrian |
| UA-DETRAC | 84000 | 25 | 578000 | Traffic |

Table 3. Datset training set comparison

## 2.5 Datasets

Since the resurgence of MOT after 2014, different general datasets have been published that aim to provide data for benchmarking object tracking. Some such datasets are the MOTChallenge datasets, KITTI, PETS2009 and UA-DETRAC, as listed in Table 3.
For pedestrians, the MOTChallenge datasets are often used. These range from the normal 2015-2020 datasets that focus on pure object tracking to more specialized datasets, such as CVPR 2020 MOTS which focuses on precise segmentation.
The KITTI dataset [10] is another often used dataset that contains hours of traffic footage, recorded from inside a car.
The PETS2009 dataset [8] contains pedestrian footage in different setups that are created to be challenging for object detection and/or tracking. UA-DETRAC [31] is a dataset with 100 video sequences, containing different traffic scenarios.

## 3 METHODOLOGY

### 3.1 Selection

The scope of this research limits it to the use of only certain algorithms, benchmarks and datasets. Based on relevance, performance, and level of establishment in research a selection of them is made.

*3.1.1 Algorithms.* This research uses YOLOv8 for object detection, and ByteTrack [34] for object tracking. YOLOv8 was selected because it is a fast and accurate one-stage detector, and it is publically accessible. The speed makes it particularly suited for online (real-time) environments, since which this study focuses on. The speed is important since it means YOLO could operate on lower-end hardware, or could keep up with higher frame rate input. It is also important to have a fast detection algorithm in live environments to leave time for the tracking algorithm to compute. For the object tracking, ByteTrack was selected because, as seen in Table 2, it performs fast enough to perform above the tested FPS range of this research. It also ranks high on the MOTChallenge rankings, uses the online (real-time) method, uses public detection (public datasets), and is open source available. The online method is the focus of this research, since this research aims to provide results for industries that use object tracking in real-time. The use of public datasets and open source are chosen since this helps verify the validity of the algorithm and allows this research to use it in a test setting.

*3.1.2 Benchmarks.* This research specifically considers MOTA, MOTP, IDP, IDR, IDF1, and FPS. MOTA and MOTP provide a general overview of the accuracy of object tracking. They paint a clear picture of accuracy, where MOTP focuses more on the accuracy of detected objects in each frame and MOTA ratios the inaccuracies of ID assignment over the whole video. IDP, IDR, and IDF1 are used to quantify the dropoff rate of correct ID assignments. IDP, similar to MOTP, tracks precision in ID assignment. It is an indicator for the amount of incorrect identifications in a video. IDR is similar to IDP, but assesses missed IDs rather than incorrectly assigned IDs. Finally, IDF1 combines IDR and IDP and indicates a more general score for incorrect IDs.

*3.1.3 Datasets.* The datasets used in this research are MOT15 [14] and MOT20 [6]. As listed in Table 3, they each contain pedestrian datasets. The purpose of this research is to focus on practical application with different frame rates. This means that the chosen datasets are those that contain slower moving objects, to increase possible variation in frame rate. If objects of higher speed are considered, objects would leave the frame faster, and thus lower frame rates could be considered before a drop-off in detection and tracking occurs. Therefore the pedestrian datasets are considered. As previously described, the MOTChallenge datasets are widely used for pedestrian object tracking. Out of those datasets, the regular datasets (MOT15 and MOT20) were chosen rather than specialized datasets (e.g. CVPR 2020 MOTS) to provide more general results. Specifically the MOT15 dataset was used since it contains a low pedestrian density, and MOT20 because it contains a high pedestrian density. A comparison between the two is valuable to show whether higher density influences performance at different frame rates. The datasets also contain videos with FPS above the frame rate tested in this research, apart from the 7 FPS PETS09-S2L1 video in the MOT15 dataset, which was therefore not included in the results of this research.

## 3.2 Setup

The used methods are combined to create a framework for measuring object tracking accuracy at different frame rates. The setup of this framework is described below.

*3.2.1 Datasets.* The MOT15 and MOT20 dataset both contain a set of training and testing videos. The training sets contain several different videos segmented into individual frames. For each frame, the ground truth is provided as a bounding box and id for each object in each frame. The test sets contain similar videos but without the ground truth. Since this research focuses on exact IDs and accuracy, pre-trained algorithms were used on only the training sets so that the ground truth could be used to calculate metrics accurately.

*3.2.2 Frame rate.* MOT15 contains videos between 7 and 30 FPS, and MOT20 contains videos of only 25 FPS. The 7 FPS video was not used in this research, so all video had a frame rate between 10 and 30 FPS. Each video is passed through object tracking algorithms at different frame rates. The considered frame rates per video range from the maximum frame possible frame rate to 0.1 FPS. The potentially considered frame rates are 10-1 FPS in steps of 1 FPS, and 1-0.1 in steps of 0.1 FPS. This is done by taking the maximum input frame rate of the video, and stepwise removing frames until the correct frame rate is achieved. This is done using the algorithm seen in Algorithm 1.

**Algorithm 1** Frame rate slicing

```
Input: $desired\_FPS$, $video$
Output: $frame\_set$, where $\frac{length(frame\_set)}{video\_time} = desired\_FPS$
  $frame\_set \leftarrow array(video.frames)$
  $step \leftarrow max\left(\frac{video.FPS}{desired\_FPS}, 1\right)$
  $segments \leftarrow floor\left(\frac{length(frame\_set)}{step}\right)$
  $result \leftarrow array()$
  $i \leftarrow 0$
  while $i < length(frame\_set)$ do
      $index \leftarrow floor(step * i)$
      $result.append(frame\_set[index])$
      $i \leftarrow i + 1$
  end while
  return $result$
```

*3.2.3 Object detection and tracking.* The segmented frames are then iteratively run through object detection and object tracking algorithms. The YOLOv8 nano model is used for object detection, and the detected objects are tracked using the ByteTrack algorithm. The output of this algorithm is stored as bounding boxes with IDs for each object, similar to the provided ground truth from the MOTChallenge dataset.

*3.2.4 Metric Computation.* Finally, the TrackEval [18] framework is used to compute CLEAR MOT metrics from the results of the object tracking compared to their ground truths. Additionally, the ID metrics are computed.

## 4 RESULTS

### 4.1 General results

Figures 1, 2, 3 and 4 show graphs of the obtained results. The figures are divided into the 0.1 to 1.0 FPS range and the 1-10 FPS range for both the MOT15 and MOT20 dataset. The considered metrics of MOTA, MOTP, IDF1, IDR, and IDP are all graphed in each figure. In the range of 0.1 to 1.0 FPS, the results do not show much variability, and the accuracies for MOTA, IDR, and IDF1 are low. In all cases, MOTP consistently scored between approximately 0.75 and 0.8 on every dataset and frame rate. This is due to the calculation of MOTP looking more into object detection than object tracking. It takes the overlap between the bounding box of a found object and its ground truth and divides this by the number of objects detected in this frame. This means that the detected objects had around 80% overlap with their true bounding box on average. This means that the object detection is working. Similarly, IDP, the precision for IDs, scores better than MOTA, IDF1, and IDR. IDP gets higher when there are

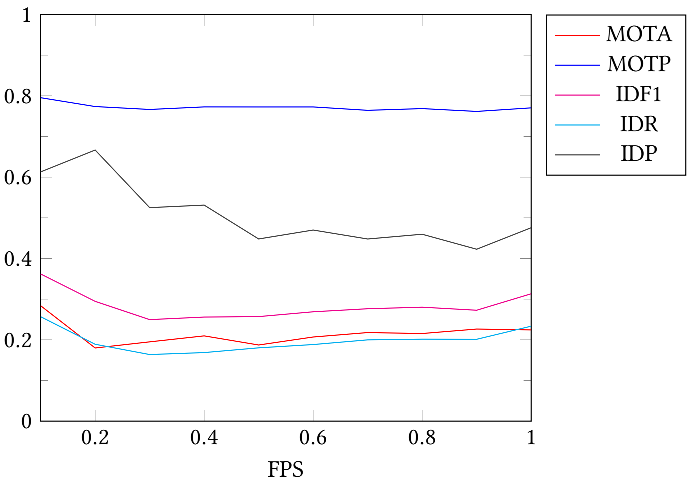

Fig. 1. Results for MOT15-1, FPS 0.1-1.0

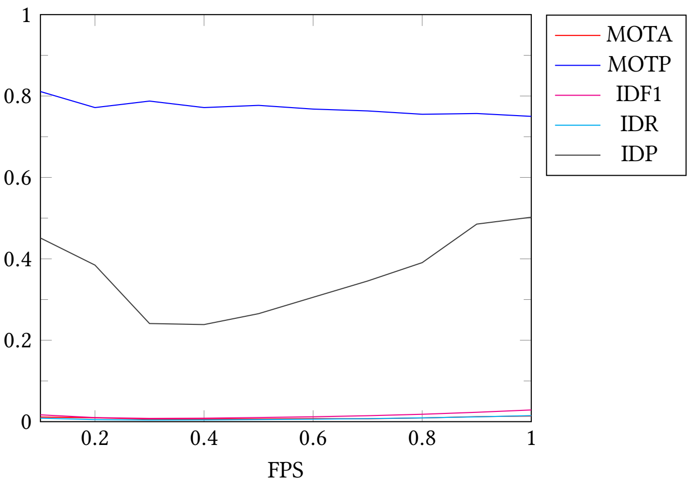

Fig. 2. Results for MOT20-1, FPS 0.1-1.0

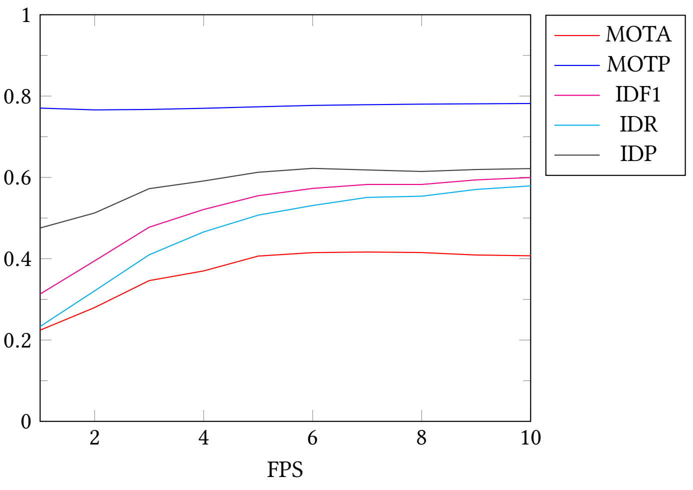

Fig. 3. Results for MOT15-10, FPS 1-10

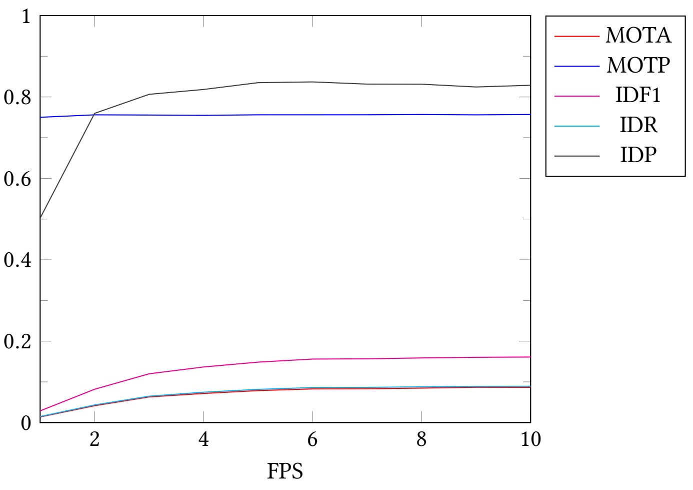

Fig. 4. Results for MOT20-10, FPS 1-10

less false positive IDs assigned (see 2.4). Thus with a with IDP scoring higher than the rest, relatively little IDs were wrongly detected as an ID. This makes intuitive sense, since with fewer frames to detect IDs in, the amount of incorrect positive detections would not necessarily rise, since the absolute amount of detections in a frame stays the same. The most interesting result is seen when observing MOTA, IDF1 and IDR in the range of 1 to 10 FPS.

*4.1.1 MOT15.* In Figure 1 the results for FPS range 0.1 to 1.0 are shown for the MOT15 dataset. We can see that the MOTA, IDR, and IDF1 scores remain low. In the range of 1 to 10 FPS, shown in Figure 3, the scores increase and seemingly plateau after 6 FPS. The IDP score is around 0.3 to 0.5 in the 0.1 to 1.0 FPS range. In the 1 to 10 FPS range however, it follows a similar curve to MOTA, IDR, and IDF1.

*4.1.2 MOT20.* In Figure 2 the results for FPS range 0.1 to 1.0 are shown for the MOT20 dataset. In this case, the MOTA, IDF1, and IDR are all near zero. The IDP however seems to again be in the range of 0.3 to 0.5. In the range of 1 to 10, shown in Figure 4, the scores go up, but less when compared to the MOT15 dataset. IDF1 reaches nearly 0.2 at most, while IDR and IDF1 reach nearly 0.1. In contrast to this, IDP scores higher here than in the MOT15 dataset. Interestingly, the same plateau effect can still be observed after 6 FPS. IDF1, IDP, IDR, and MOTA all increase until reaching 6 FPS, where they stay nearly constant.

## 5 RELATED WORK

In a 2017 paper, Kiani & Fagg already mention frame rate as a limiting factor to object tracking [11]. They addressed this issue by introducing a dataset of various real world scenarios recorded at a frame rate of 240 FPS. As a comparison, they sliced their 240 FPS footage down to 30 FPS. To make a fair comparison, they also introduced motion blur to the lower FPS footage. This was necessary since the high-speed cameras required to record videos at 240 FPS have less motion blur than regular cameras. While Kiani & Fagg conclude that simple trackers perform better than complex trackers on high FPS. They also show that out of a set of the best trackers they evaluated, 8 out of 30 instances were not able to track in real-time.

Sudasingha et al. tackle a different issue related to frame rate. In their 2019 paper about frame rate related to human pose estimation [30], they mention that a real-time video could not be processed by a pose estimation system, due to the time it takes to process each frame. They propose a system in which some frames are dropped to allow for real-time processing, and where the dropped frames are processed by a faster pose generator. The generated and estimated frames are subsequently reassembled into a video output. Sudasingha et al. measured accuracy by using a generated frames per detected frame (GF/DF) metric to compare reduction in error when their generative pose estimator (GPE) is used to when it is not used. They find that the GPE is suitable for various frame rates, observing a 30-35% reduction in error on 4-6 FPS. Notably, on lower frame rate configurations of 0.5-1 FPS for the pose estimator, improvements to the accuracy are observed (40-50% reduction in error).

Saito et al. in their 2023 paper look at frame rate related to video-based vehicle counting [27]. They use the YOLOv4 object detection algorithm trained on the COCO data-set, combined with the DeepSORT object tracking algorithm. This setup tracks vehicles at an intersection, and counts them based on the behavior of each vehicle's movement. The video stream of 30 FPS was sliced into videos of 1, 2, 3, 5, 6, 10 and 15 FPS. Additionally, they used a method based on average images for frame interpolation to generate a 60 FPS video out of the 30 FPS video. They show that interpolation reduces the amount of false positives vehicle counts, while the amount of false negatives stays the same. For the lower FPS, the number of missing vehicles decreased as frame rate went up, although the amount seems to flatten out after 5 FPS. This is however difficult to confirm, since after 6 FPS the measurements jump to 10, 15 and subsequently 30 FPS.

For object detection rather than object tracking, frame rate has also been researched. Padilla et al. wrote in their 2021 paper about the lack of object detection evaluation at video level [23]. They proposed spatio-temporal tube average precision (STT-AP) as an extension of the average precision (AP) metric. STT-AP is defined as a collection of bounding boxes of the same object in a video, concatenated after each other to form a spatio-temporal tube. The overlap between the tube found by the model and the ground truth tube can than be calculated as an intersection over union (IOU) to calculate the results of this metric. While this is an interesting method, it seems to simply offer an alternative to regular MOT metrics. This is because it involves tracking objects over different frames, while taking into account the total area (or in this case volume) of correctly identified bounding boxes, similar to MOT metrics like MOTA or MT. It does not offer further insight into varying frame rate in object tracking.

## 6 CONCLUSION AND DISCUSSION

### 6.1 Conclusion

This study set out to investigate how varying frame rate affects accuracy in object tracking. It used state-of-the-art object detection and object tracking together with established pedestrian datasets for this investigation. These datasets were sliced in varying frame rates, and measured using relevant established benchmarks to assess the resulting accuracy of object tracking. The results showed that after 6 FPS little to no variance was seen in the accuracy of object tracking. This seems to suggest that the optimal range for pedestrian tracking using the ByteTrack framework is at 6 FPS or less. In the context of computational limitations and sustainable AI, this is valuable since it suggests that for this purpose any frame rate above 6 FPS is wasting additional resources without gaining additional value in return.

### 6.2 Discussion

Some alterations could be made to improve the methodology in this work. For one, the frame rate slicing could be altered. Currently, the highest frame rate is considered and frames are removed from this until the desired frame rate is reached, as seen in Algorithm 1. This simulates the desired frame rate rather than achieving it. In cases where the maximum frame rate is 10 FPS (such as some of the MOT15 videos), this means that for instance a frame rate of 9 FPS is achieved by skipping 1 in every 10 frames. The distance between the remaining frames however is still similar. Recording multiple videos at different frame rates to get around this problem would not be fair, since then the comparison might be off by more or fewer objects being in a single frame. A possible solution could be to take higher frame rate videos as source materials, as done in [11]. At source frame rates at least 2 times higher than the maximum tested frame rate, the time between any two consecutive frames would always change when using the frame removal method.

Additionally, more analysis could be done to assess how much the used FPS lowering methods differ from the real world. In [11] for instance, the authors noted that high frame rate cameras tend to have less motion blur. Thus they introduced custom motion blur on the sliced high FPS footage to mimic the effect. This could however still differ from real low FPS motion blur. Furthermore, the use of only a pedestrian dataset limits the scope of the results obtained from this research. Pedestrians in the MOTChallenge datasets tend to move slower compared to for instance the moving vehicles in the KITTI dataset. Since object tracking also has wide applications in areas such as self driving cars and traffic situations, more research should be done on those datasets to test if the conclusions from this work transfer to higher speed as well as differently sized objects such as cars.

### 6.3 Future work

One valuable addition to this research would be increasing the quantity of the tested items. More datasets could be tested to improve the

breadth of the results. More algorithms could be tested to conclude a wider part of the object tracking spectrum. A wider range of FPS could be tested, to see if the observed plateau effect after 6 FPS persists further beyond 10 FPS.
As described, the results of this research could potentially be useful for specific industries using object tracking. More research could be done looking into frame rate variation on industry-specific datasets, similar to [27].
The object tracking in this research relied on object detection done by the YOLOv8 algorithm. While this algorithm is well-suited for higher FPS detection due to its speed, many MOT challenges use pre-computed object detection, by providing the exact bounding boxes to the tracking algorithm. This research focused on more realistic circumstances by detecting objects at run time to simulate the circumstances in the field. Still, a comparison between different tracking algorithms with pre-computed bounding boxes could be valuable to have a more pure evaluation between algorithms.
A final interesting note is that for the results and operation of this research, technically slower (and more accurate) algorithms could also be used. Since only the input frame rate is varied, and the actual runtime is not considered, potentially interesting results could lie in testing lower frame rate for slower algorithms. For this research that was not considered, since it focuses on providing results that could have more practical implications. That is, considering real-time environments, an algorithm that by default runs slower than the tested input frame rate would likely not be implemented in practice. It could not keep up with the input feed and would thus perform poorly in real-time.

## 7 ACKNOWLEDGEMENTS

Thanks goes out to supervisor Rob Bemthuis who helped guide this research, and to the interviewees working with object detection in the industry for providing datasets and useful insights.

This research is part of the ECOLOGIC project, which was funded by the Dutch Ministry of Infrastructure and Water Management and TKI Dinalog (case no. 31192090).

## REFERENCES

[1] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. 2022. BOT-SORT: Robust Associations Multi-Pedestrian Tracking. *arXiv (Cornell University)* (2022). https://doi.org/10.48550/arxiv.2206.14651
[2] Ohay Angah and Albert Y. Chen. 2020. Tracking multiple construction workers through deep learning and the gradient based method with re-matching based on multi-object tracking accuracy. *Automation in Construction* 119 (2020), 103308. https://doi.org/10.1016/j.autcon.2020.103308
[3] Keni Bernardin and Rainer Stiefelhagen. 2008. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing* 2008 (2008), 1–10. https://doi.org/10.1155/2008/246309
[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-End Object Detection with Transformers. In *Computer Vision – ECCV 2020*. Cham, 213–229. https://doi.org/10.48550/arXiv.2005.12872
[5] Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, Luigi Troiano, Roberto Tagliaferri, and Francisco Herrera. 2020. Deep learning in video multi-object tracking: A survey. *Neurocomputing* 381 (2020), 61–88. https://doi.org/10.1016/j.neucom.2019.11.023
[6] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé. 2020. MOT20: A benchmark for multi object tracking in crowded scenes. (2020). https://doi.org/10.48550/arXiv.2003.09003 Accessed 10 June 2024.
[7] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. 2009. The Pascal Visual Object Classes (VOC) challenge. *International journal of computer vision* 88, 2 (2009), 303–338. https://doi.org/10.1007/s11263-009-0275-4
[8] J. Ferryman and A. Shahrokni. 2009. PETS2009: Dataset and challenge. In *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*. 1–6. https://doi.org/10.1109/PETS-WINTER.2009.5399556
[9] Kent Gauen, Ryan Dailey, John Laiman, Yuxiang Zi, Nirmal Asokan, Yung-Hsiang Lu, George K. Thiruvathukal, Mei-Ling Shyu, and Shu-Ching Chen. 2017. Comparison of Visual Datasets for Machine Learning. In *2017 IEEE International Conference on Information Reuse and Integration (IRI)*. 346–355. https://doi.org/10.1109/IRI.2017.59
[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 3354–3361. https://doi.org/10.1109/CVPR.2012.6248074
[11] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. 2017. Need for Speed: A Benchmark for Higher Frame Rate Object Tracking. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.48550/arXiv.1703.05884
[12] Hilke Kieritz, Wolfgang Hübner, and Michael Arens. 2018. Joint Detection and Online Multi-object Tracking. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 1540–15408. https://doi.org/10.1109/CVPRW.2018.00195
[13] Hei Law and Jia Deng. 2018. CornerNet: Detecting Objects as Paired Keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*. https://doi.org/10.48550/arXiv.1808.01244
[14] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. 2015. MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking. (2015). https://doi.org/10.48550/arXiv.1504.01942 Accessed 10 June 2024.
[15] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. 2017. Focal Loss for Dense Object Detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.48550/arXiv.1708.02002
[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*. Cham, 740–755. https://doi.org/10.1007/978-3-319-10602-1_48
[17] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. 2016. SSD: Single Shot MultiBox Detector. In *Computer Vision – ECCV 2016*. Cham, 21–37. https://doi.org/10.48550/arXiv.1512.02325
[18] Jonathan Luiten and Arne Hoffhues. 2020. TrackEval. https://github.com/JonathonLuiten/TrackEval. https://doi.org/10.1007/s11263-020-01375-2
[19] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip H. S. Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. 2020. HOTA: A Higher Order Metric for Evaluating Multi-object Tracking. *International journal of computer vision* 129, 2 (2020), 548–578. https://doi.org/10.1007/s11263-020-01375-2
[20] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. 2021. Multiple object tracking: A literature review. *Artificial Intelligence* 293 (2021), 103448. https://doi.org/10.1016/j.artint.2020.103448
[21] Huizi Mao, Xiaodong Yang, and William J Dally. 2019. A delay metric for video object detection: What average precision fails to tell. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 573–582. https://doi.org/10.48550/arXiv.1908.06368
[22] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. 2016. MOT16: A Benchmark for Multi-Object Tracking. (2016). https://doi.org/1603.00831
[23] Rafael Padilla, Wesley L. Passos, Thadeu L. B. Dias, Sérgio L. Netto, and Eduardo A. B. Da Silva. 2021. A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. *Electronics* 10, 3 (2021), 279. https://doi.org/10.3390/electronics10030279
[24] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon Emissions and Large Neural Network Training. https://doi.org/10.48550/arXiv.2104.10350
[25] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 779–788. https://doi.org/10.1109/CVPR.2016.91
[26] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. 2016. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*. 17–35. https://doi.org/10.48550/arXiv.1609.01775
[27] Koji Saito, Sho Takahashi, and Toru Hagiwara. 2023. A Note on Improvement of Multi Object Tracking by Frame Interpolation for Intersection Traffic. In *2023 IEEE International Conference on Consumer Electronics (ICCE)*. 1–2. https://doi.org/10.1109/ICCE56470.2023.10043581
[28] Juan Terven, Diana-Margarita Córdova-Esparza, and Julio-Alejandro Romero-González. 2023. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *Machine Learning and Knowledge Extraction*

5, 4 (2023), 1680–1716. https://doi.org/10.48550/arXiv.2304.00501

[29] Aimee Van Wynsberghe. 2021. Sustainable AI: AI for sustainability and the sustainability of AI. *AI and ethics* 1, 3 (2021), 213–218. https://doi.org/10.1007/s43681-021-00043-6

[30] Madhawa Vidanpathirana, Imesha Sudasingha, Jayan Vidanapathirana, Pasindu Kanchana, and Indika Perera. 2019. Tracking and frame-rate enhancement for real-time 2D human pose estimation. *The visual computer/ The visual computer* 36, 7 (2019), 1501–1519. https://doi.org/10.1007/s00371-019-01757-9

[31] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. 2020. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *Computer Vision and Image Understanding* 193 (2020), 102907. https://doi.org/10.1016/j.cviu.2020.102907

[32] Adi Wibowo, Joga Dharma Setiawan, Hadha Afrisal, Anak Agung Sagung Manik Mahachandra Jayanti Mertha, Sigit Puji Santosa, Kuncoro Budhi Wisnu, Ambar Mardiyoto, Henri Nurrakhman, Boyi Kartiwa, and Wahyu Caesarendra. 2023. Optimization of Computational Resources for Real-Time Product Quality Assessment Using Deep Learning and Multiple High Frame Rate Camera Sensors. *Applied System Innovation* 6, 1 (2023). https://doi.org/10.3390/asi6010025

[33] Bo Wu and Ram Nevatia. 2006. Tracking of multiple, partially occluded humans based on static body part detection. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06).* 951–958. https://doi.org/10.1109/CVPR.2006.312

[34] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. 2022. Bytetrack: Multi-object tracking by associating every detection box. In *European conference on computer vision.* 1–21. https://doi.org/10.48550/arXiv.2110.06864

[35] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. 2019. Objects as points. *arXiv preprint arXiv:1904.07850* (2019). https://doi.org/10.48550/arXiv.1904.07850

[36] Haidi Zhu, Haoran Wei, Baoqing Li, Xiaobing Yuan, and Nasser Kehtarnavaz. 2020. A Review of Video Object Detection: Datasets, Metrics and Methods. *Applied Sciences* 10, 21 (2020). https://doi.org/10.3390/app10217834

[37] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2023. Object Detection in 20 Years: A Survey. *Proc. IEEE* 111, 3 (2023), 257–276. https://doi.org/10.1109/JPROC.2023.3238524