drones

MDPI

Article

Towards Real-Time On-Drone Pedestrian Tracking in 4K Inputs

Chanyoung Oh [1,*], Moonsoo Lee [2] and Chaedeok Lim [2]

[1] Department of Software, Kongju National University, Cheonan 31080, Republic of Korea
[2] Air Mobility Research Division, Electronics and Telecommunications Research Institute, Daejeon 34129, Republic of Korea; mslee@etri.re.kr (M.L.); cdlim@etri.re.kr (C.L.)
* Correspondence: cyoh@kongju.ac.kr

Abstract: Over the past several years, significant progress has been made in object tracking, but challenges persist in tracking objects in high-resolution images captured from drones. Such images usually contain very tiny objects, and the movement of the drone causes rapid changes in the scene. In addition, the computing power of mission computers on drones is often insufficient to achieve real-time processing of deep learning-based object tracking. This paper presents a real-time on-drone pedestrian tracker that takes as the input 4K aerial images. The proposed tracker effectively hides the long latency required for deep learning-based detection (e.g., YOLO) by exploiting both the CPU and GPU equipped in the mission computer. We also propose techniques to minimize detection loss in drone-captured images, including a tracker-assisted confidence boosting and an ensemble for identity association. In our experiments, using real-world inputs captured by drones at a height of 50 m, the proposed method with an NVIDIA Jetson TX2 proves its efficacy by achieving real-time detection and tracking in 4K video streams.

Keywords: on-device deep learning; unmanned aerial vehicle (UAV); drone; real-time object tracking

1. Introduction

Interest in unmanned aerial vehicles (UAVs) has been on the rise in recent years, with applications spanning a wide range of fields such as aerial photography [1], surveillance [2,3], search and rescue [4,5], inspection of infrastructure [6], traffic monitoring [7], counting animals [8], and so forth. One of the key requirements for these applications is the ability to detect and track objects of interest. Object detection and tracking (simply, object tracking) is a fundamental task in computer vision that consists of localizing objects of interest over time in video streams. With the increasing availability of high-quality cameras and the proliferation of video data, the need for accurate and efficient object tracking has become more important than ever. In recent years, the field of object tracking has undergone a significant revolution thanks to the advent of deep learning-based algorithms [9–13] and the availability of huge datasets [14,15].

Object tracking is particularly crucial for UAVs such as drones because of their ability to capture videos and images from unique viewpoints, as shown in Figure 1. UAVs are often used in applications where it is not feasible or safe for humans to operate. Object tracking can also aid in real-time decision-making by enabling UAVs to avoid obstacles, drive autonomously, or follow moving targets [16,17]. Therefore, accurate and efficient object tracking is critical for maximizing the effectiveness and safety of UAVs. In this paper, we consider pedestrians as the target object. Pedestrian tracking is a common application of object tracking, which is employed to precisely determine the spatial coordinates of individuals and subsequently monitor their movement trajectories. In various contexts, including crowd management and surveillance, pedestrian tracking plays a crucial role.

check for updates

Citation: Oh, C.; Lee, M.; Lim, C. Towards Real-Time On-Drone Pedestrian Tracking in 4K Inputs. *Drones* **2023**, *7*, 623. https://doi.org/10.3390/drones7100623

Academic Editors: Giancarlo Rufino and Claudia Conte

Received: 21 August 2023
Revised: 4 October 2023
Accepted: 4 October 2023
Published: 6 October 2023

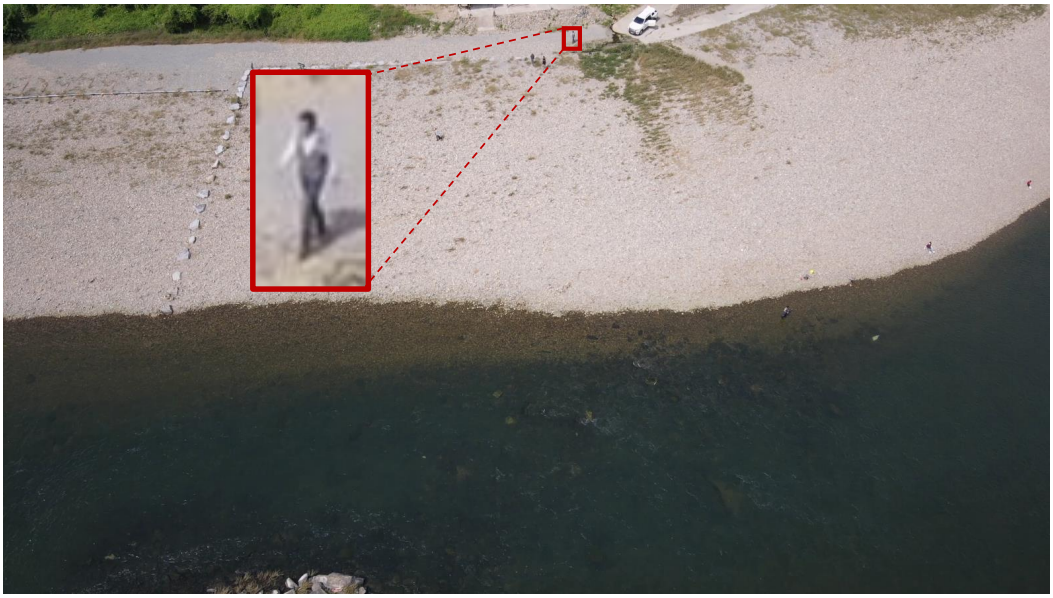

Figure 1. A drone-captured image at the height of 50 m. To detect and track pedestrians from a scene in high altitude, typically, ultra high-definition (4K) input is required.

Despite the potential benefits of object tracking on UAVs, there are several challenges that must be addressed to achieve robust and reliable tracking performance. One major challenge is the need for online real-time processing of tasks, while drones have limited computing power. Since wireless network connections are not always guaranteed in the air, drones must rely on onboard mission computers to perform their tasks in real time, even though their workloads, such as deep learning algorithms, demand high computing power so as to ensure the safety and reliability of operation, as well as enabling autonomous and intelligent applications. While certain studies, such as [18], leverage the 5G network and server-level systems to attain real-time tracking on UAVs, it is crucial to note that the applicability of 5G networks in aerial contexts might be limited. This limitation arises due to the fact that commercial 5G networks are primarily designed for terrestrial communications. Therefore, the capability of on-device online real-time processing holds significant meaning in various drone scenarios such as obstacle avoidance. In many cases, the real-time constraint is given by 33 ms, with the prevalent adoption of a 30 Hz frequency in video streaming applications. In addition, target objects in drone-captured images are often too small due to the high altitude at which the images are captured (e.g., 50–100 m). To address this issue, the resolution of the camera needs to be increased, which, in turn, increases the workload for analysis. This can make it even harder to resolve the challenge of high computational demands for real-time processing. On the other hand, the fast motion of vehicles (e.g., rotation and vibration) can cause the tracker to lose track of objects as it rapidly changes the scene, making object tracking more challenging.

In current practices, object tracking typically follows the tracking-by-detection paradigm in which object detection is performed for each frame and the outputs of detection, along with optionally the intermediate results of the detection process, are utilized to track the detected objects. This strategy often suffers from huge computational demand in case real-time processing is required. On the other hand, traditional object tracking algorithms such as minimum output sum of squared error (MOSSE) [19] demonstrate relatively low tracking latency. However, these algorithms lack the ability to associate detected objects with the objects being tracked since they simply trace manually provided regions from users in a video. In the context of UAVs, object tracking deals with real-time input streams, and objects can appear at any time. Therefore, the association of detection and tracking that maintains comparable tracking accuracy and meets real-time constraint becomes essential for effective object tracking on UAVs.

The goal of this work is to develop a system that enables online real-time object tracking on drones by bridging the gap between the traditional trackers and the modern tracking-by-detection trackers. We introduce an association strategy that combines a time-consuming deep learning-based detection algorithm with a fast tracking algorithm. The key aspect of this strategy is to determine when and where to perform detection and tracking procedures, respectively, while also effectively combining the results from both. This paper presents our approaches for real-time pedestrian tracking on drones as a case study, which aims to detect and track tiny objects from a bird's eye view. The main idea is *intermittent detection and parallel execution*, which exploits all the processors in the mission computer. It consists of an *input slicing and stitching* technique that reduces the detection latency by leveraging locality. We also propose *an ensemble for identity association* that combines lightweight features to efficiently distinguish tiny objects. Then, we employ *tracker-assisted confidence boosting* in which the problem of insecure detection confidence due to motion blur and the small size of objects are mitigated.

Experiments show that a drone can track pedestrians from a 4K onboard camera in real time without the need for external servers or clouds.

Overall, this work makes the following contributions:

- It develops a real-time pedestrian tracker for the onboard mission computer, which takes as the input 4K aerial video streams. The fundamental ideas are to determine when and where to execute detection and tracking algorithms and to combine the results from them effectively.
- It proposes a novel tracker-assisted confidence boosting algorithm to enhance the detection accuracy.
- It empirically demonstrates the efficacy of the proposed methods on real-world aerial videos, which are captured by drones at the height of 50 m.
- To the best of our knowledge, this paper is the first work that enables real-time on-drone tracking for 4K aerial inputs.

The rest of this paper is organized as follows. In Section 2, we briefly summarize related studies and compare our work with them. Then, Section 3 illustrates an overview of the proposed real-time pedestrian tracker for drones, which is followed by the key approaches. The experimental results are shown in Section 4, and we conclude in Section 5.

2. Related Works

2.1. Single Object Tracking (SOT)

Single object tracking (SOT) is the most classic type of tracking. It finds a target specified in the first frame (e.g., bounding box) in the subsequent frames. Representative algorithms include KCF [20], CSRT [21], and MOSSE [19].

Kernelized correlation filters (KCF) [20] is an object tracking algorithm that aims to achieve high-speed tracking performance. It employs a set of filter templates and a kernel function and in frequency domain to map the features of the object into a high-dimensional space. The discriminative correlation filter with channel and spatial reliability tracker (CSRT) [21] combines the correlation filter-based tracking and spatial reliability-based tracking approaches to track objects with high accuracy. MOSSE [19] models the target appearance using an adaptive correlation filter and determines the new position of the target as the location where the output of convolution has the maximum value. The confidence of tracking is detected using peak-to-sidelobe ratio (PSR).

These SOT algorithms are simple and fast, but they suffer from erroneous update policies. For instance, when the target object is occluded, these algorithms may update the feature of the obstacle, causing them to fail in tracking the actual desired object. This happens because these algorithms update the information inside the bounding box over time. Furthermore, these algorithms lack the ability to associate detected objects with tracks, making it difficult to distinguish between a new object and an object that has reappeared after occlusion.

One may consider using person re-identification algorithms [22–25] as an alternative to SOT, as they are capable of identifying the same individual across different frames or viewpoints. However, re-identification tasks require significant computational resources, making it infeasible to implement for on-drone real-time processing as well.

2.2. Multiple Object Tracking (MOT)

Multiple object tracking (MOT) aims to track multiple objects simultaneously. The most common strategy used in recent years is tracking-by-detection, and the representative algorithm for this approach is SORT [11]. SORT consists of detection, estimation, and data association processes. During the detection process, YOLO-based detectors [26] are often adopted, while Faster R-CNN [27] was used in the original SORT [11], as they show outstanding detection accuracy and efficiency. A variant of YOLO has also been proposed for drone video [28]. Estimation uses Kalman filter to predict the position and the velocity of objects, and data association uses the Hungarian algorithm to match detection and prediction with intersection over union (IoU) as the metric. However, Kalman filter and IoU-based approaches are vulnerable in on-drone tracking scenarios where objects' locations in the scene may be changed rapidly due to the drone's movement, such as rotation.

Recently, MOT algorithms that utilize deep features have been attempted, such as DeepSORT [12] and FairMOT [9]. DeepSORT utilizes appearance information through deep feature extraction, as well as IoU, and reduces identity switching by 45%. FairMOT argues that MOT's performance is affected by re-identification tasks, highlighting the problems of existing trackers with re-identification, such as dependency on primary tasks (i.e., detection), shallow feature dimensions, and discrimination of features. To solve these problems, one-shot trackers were proposed to perform detection and tracking simultaneously [29], but the naive approach caused performance degradation. FairMOT utilizes an encoder-decoder backbone network with deep layer aggregation to combine high-level and low-level features appropriately.

On the other hand, some works leverage the relationship among objects rather than relying on the IoU and appearance features [10,30–32]. GSM [10], for example, proposed a graph similarity model that considers the relative position of objects. In this model, graph vertices represent each object and its neighbors' appearance features, and edges represent relative positions. Appearance features are extracted from CNN as in other algorithms, and relative positions are embedded as attention [33]. Graph matching is then performed. Lost objects due to occlusion or detection miss can be estimated by reversing the relative position with neighbors and taking the average.

However, most recent object trackers require detection at every frame, which makes real-time processing of high-resolution inputs impossible on drones.

2.3. Drone Datasets

With the increasing interest in drones and their applications, there have been attempts to establish drone-captured datasets to boost the development of various UAV applications. Since the drone's tasks are typically performed with a unique view point (i.e., bird's eye view), employing a traditional dataset [34] is insufficient.

A series of VisDrone datasets [14,35] provide large-scale images and videos captured by drones, specifically designed for drone-related computer vision tasks. However, the data vary vastly in quality and size.

Recently, the Electronics and Telecommunications Research Institute (ETRI) released a new drone dataset called the DNA+Drone Dataset [15], which includes 4K high-resolution images and videos captured by drones in outdoor environments. The dataset covers a specific range of heights (i.e., 50–150 m) and angles (45 and 90 degrees), making it a valuable resource for researchers working on drone-related computer vision tasks.

3. Design and Implementation

This section offers the proposed design and implementation of a real-time on-drone pedestrian tracker that takes as the input 4K aerial videos. We first present the overall structure and then give the key components.

3.1. Overview

3.1.1. Target Hardware Architectures

In current drone systems, the flight controller (FC), such as PixHawk, is responsible for the drone's flight operations, while complex tasks such as object detection and tracking are performed by a separate mission computer (MC). To meet the computational requirements of modern drone applications, the integration of a central processing unit (CPU) and a graphics processing unit (GPU) within the MC has emerged.

In this paper, we make the assumption that the MC includes both a CPU and a GPU, along with a unified memory system. This configuration is commonly found in mobile system-on-chips (SoCs) such as NVIDIA Jetson, Qualcomm Snapdragon, and Samsung Exynos, which fulfill the requirements for our proposed approach. In this paper, we test NVIDIA Jetson TX2 and AGX Xavier as the reference platforms.

3.1.2. Overview of the Proposed Tracker

Figure 2 shows the workflow and major components of the proposed real-time on-drone pedestrian tracker. The recent trend in object tracking is to follow the tracking-by-detection strategy. Our approach follows the strategy as well, but with a key difference.

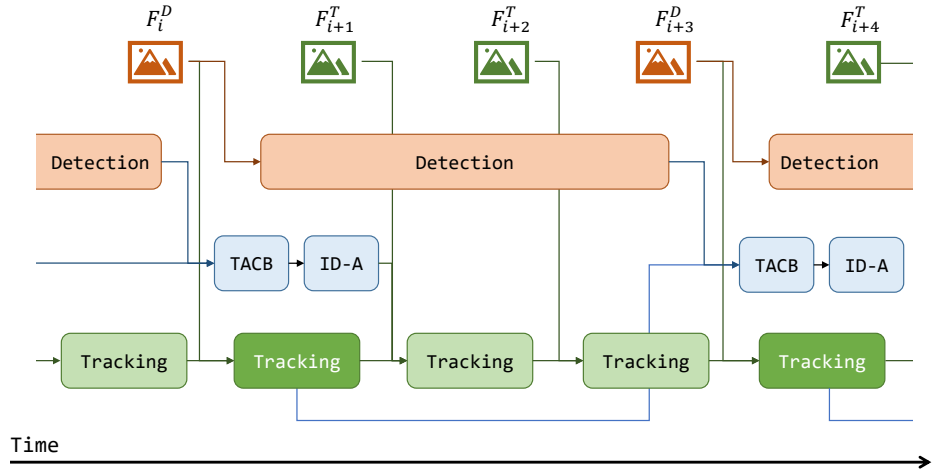

Figure 2. The overall structure and workflow of the proposed real-time on-drone pedestrian tracker. Detection and tracking are performed in parallel on a CPU and a GPU. The tracker-assisted confidence boosting (TACB) is utilized to compensate for the confidence of the detected candidates, and the compensated outputs of the detection are associated with the tracks (ID-A: identity association).

Rather than performing the time-consuming detection process on every frame, we perform it intermittently using only a small set of input frames (i.e., the frames marked in orange color in Figure 2). On the other hand, the tracks are updated in every frame by our tracking algorithm, by which a drone can track target objects in real time. Note that the processing time of detection is proportional to the input frame size, while the processing time of tracking is typically proportional to the number of objects being tracked.

One of the key components of the proposed real-time on-drone tracker is the identity association between the tracks and the detected objects. In this procedure, the objects detected during the detection phase are classified into two groups: new objects and objects that are already being tracked. We determine the object as a new object in case it appears in three consecutive detection frames and it does not match any existing tracking objects.

Target objects in images captured by drones at a high altitude are typically very small, which makes detection accuracy low. We resolve this problem by proposing a novel tracker-assisted confidence boosting algorithm (Section 3.2.3). It improves the reliability of detection.

It should be noted that this paper focuses on detecting and tracking people as objects, and no other types of objects are considered. However, the underlying concepts and ideas presented can be applied to track other types of objects in aerial inputs as well. Some interesting problems to explore include animal tracking, where we have the capability to monitor and assess the dynamics of events such as horse racing.

In the following sections, we present the key techniques in more detail.

3.2. Implementation Details

A major challenge for enabling real-time on-drone pedestrian tracking is the long latency of performing detection on high-resolution drone-captured inputs. While recent advancements in convolutional neural network (CNN)-based detection have significantly improved accuracy, it remains challenging to employ such computationally intensive methods on drones with limited resources. Sometimes, a YOLO detector [26] may take more than one second to process a 4K image, even with a lightweight backbone. Typically, an input video stream operates at a 30 Hz update rate, which implies that achieving a latency of 33 ms is necessary to meet real-time performance requirements. This work adopts YOLO as the detector as well.

On the other hand, the computational cost of most tracking algorithms is not dependent on the input size but rather on the number of objects being tracked. In our observation, the latency of MOSSE tracking algorithms [19] is typically within a few milliseconds on a drone's mission computers, which makes it suitable for real-time processing. We employ the MOSSE tracker [19], a lightweight SOT algorithm, as our tracking algorithm. As mentioned in Section 2, MOSSE suffers from wrong feature update. However, it is the efficiency rather than robustness that is crucial in our scenario (i.e., input update rate of 30 Hz), as tracking only lasts for 5–20 frames during the period of detection. The lost tracks can be easily restored by detection.

3.2.1. Intermittent Detection and Parallel Execution

This work adopts the tracking-by-detection approach, which has become a popular trend in object tracking. It first detects target objects in the input frame and then performs tracking using the output of the detection phase. The proposed work is distinguished from previous works in that we perform the detection phase intermittently for only a small set of input frames. Since the latency of the tracking algorithm is much lower than that of the detection algorithm, reducing the occurrence of run of detection is crucial for achieving real-time processing.

Then, our tracker runs in parallel, in which all input frames are used to track the target objects. Since the latency of tracking is much lower than the real-time constraint of less than 33 ms, the tracking results can be updated in every frame, achieving real-time performance.

As we mentioned in Section 3.1, our target MC incorporates both a CPU and a GPU. Refer to Section 4 for the specific platform we used. It allows for efficient processing of computationally demanding tasks on the GPU, while the CPU handles lighter tasks. Specifically, the time-consuming detection phase is executed on the GPU, while the tracker operates in parallel on the CPU.

In this scenario, we need to resolve the major drawback: There is a time gap between the detected objects and the current tracks. Let F_i^T denote the i-th input frame, which is fed to the tracking phase, and F_i^D is the i-th input frame used by both detection and tracking phases. As shown in Figure 2, for example, the detection process with F_i^D completes when F_{i+3}^D is arriving. In practice, the gap is much longer.

To associate the detected outputs with tracks in the right way, the input frame of detection (e.g., F_i) and the resultant tracks are propagated. In Figure 2, the output of

tracking, marked in a dark green box, is delivered to both the next frame (F_{i+2}) and the future frame (F_{i+3}).

3.2.2. Input Slicing and Output Stitching

Even if we are able to hide the latency of detection by parallel execution, the long processing time can still negatively impact the accuracy of tracking. As the gap between the detections increases, the distance between the same object in different frames can become farther apart, making it difficult to associate them as the same object.

In order to minimize the detection latency, we divide the 4K input into smaller patches and group them into a batch before performing detection. Note that while YOLO detectors often use 448×448 or 608×608 pixels as their input resolution, the input size can vary. This strategy potentially reduces the processing time since the convolution operations in the YOLO detector, which are the most time-consuming parts, take advantage of the locality by reusing the convolution filters for patches in a batch once they are loaded into a fast scratchpad memory or cache. Sometimes, one may adopt a crop selection strategy to reduce detection time, where only a cropped region of the image is used for detection. However, this approach is typically suitable when the image size is similar to the input size, as it may result in the loss of information outside the cropped region.

Note that dividing the input into as many patches as possible does not necessarily lead to a better latency, as it can result in longer processing times due to the need to consider objects at the boundaries of patches. To prevent missing objects at the patch boundaries, the patches need to overlap with each other, typically by the size of the target objects.

At the end of the detection phase, the outputs of the patches in the batch are stitched together, and the coordinates of each resulting candidate object in the patches are scaled to correspond to the position in the original 4K input. The same object found in multiple patches is integrated in this procedure by non-maximum suppression. Figure 3 illustrates the overall procedure.

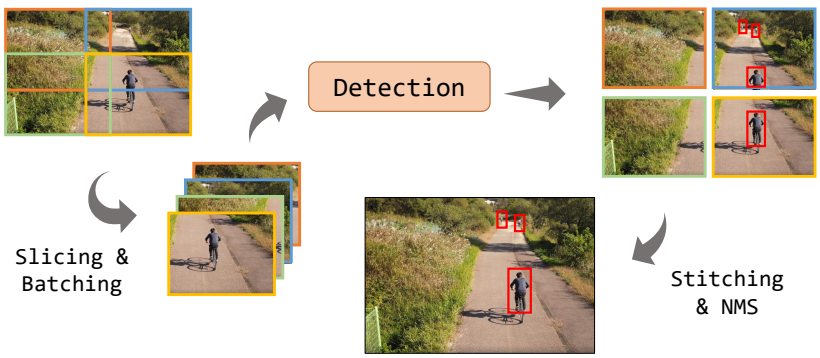

Figure 3. The overview of input slicing and output stitching. A large 4K input is divided into several patches and then grouped into a batch. The outputs of detection with the batched patches are stitched at the end of detection. This procedure is implemented in a CUDA kernel and runs in parallel on a GPU.

In our implementation, input slicing and output stitching are implemented as a CUDA kernel, which runs on a GPU. It is worth noting that the computation of this strategy does not depend on the content of the input frame.

3.2.3. Tracker-Assisted Confidence Boosting

Finding pedestrians in drone-captured images from high altitudes can be challenging due to the small size of the objects. In particular, detectors that employ very deep features are not suitable for real-time on-drone scenarios due to their long processing time and computation and battery limitations of drones.

We observed that most lightweight detectors are usually capable of finding candidates of target objects, but they lack confidence. Consequently, the candidates are rejected as their

confidence score (x_{conf}) is lower than the threshold (θ_{conf}) even if there are target objects in fact.

We propose tracker-assisted confidence boosting to overcome this problem. Given our knowledge of the locations of target objects being tracked, we are able to compare the detected candidates with the tracks, allowing us to refine the detection by applying a soft threshold (θ_{tacb}). As previously mentioned, the footprints of tracks in F^D are propagated. After detection is completed, the propagated tracks are compared to the detection results by IoU. If a track and a detection overlap, the candidate can be considered a true detection only if its confidence meets a moderate threshold for boosting.

3.2.4. Identity Association for Drone-Captured Inputs

In the tracking-by-detection strategy, the critical step is assigning the candidate objects detected in each frame to the objects being tracked. This identity association problem can be solved by a linear assignment algorithm, the Hungarian algorithm, in many cases. This work also adopts the popular algorithm used in SORT [11]. To solve the association problem with the linear assignment algorithm, we define a cost matrix $\mathcal{M}$. Let $D^i = \{d^i_j\}_{j=1}^{N^i_D}$ and $T^i = \{t^i_k\}_{k=1}^{N^i_T}$ denote the set of detections and tracks in F_i, where N^i_D and N^i_T are the number of detections and tracks in F_i, respectively. Then, the element $m_{j,k}$ in $\mathcal{M}$ is defined as:

$$m_{j,k} = \mathcal{C}(d_j, t_k) \tag{1}$$

Most other works associate detections D^{i-1} and D^i in two consecutive input frames F_{i-1} and F_i. To narrow the time gap between the frames, they typically estimate the target's velocity and location using Kalman filtering [10–12]. In contrast, our approach does not require estimating the target's velocity and location since we propagate the tracking result and directly compare it with the detections. This reduces unnecessary computation and simplifies the tracking pipeline. Thus, we build the cost matrix $\mathcal{M}$ with the detections and tracks in the same frame as Equation (1).

The most commonly used cost matrix is based on the intersection-over-union (IoU) distance function. While more advanced distance metrics, such as the graph similarity model [10], may be of interest to some, their adoption in on-drone environments is infeasible due to the limited computational resources.

In this paper, we propose an ensemble method for identity association that takes into account the characteristics of on-drone environments. The use of IoU distance assumes slow and gradual movement of target objects, which is not always the case for drones. Drones may exhibit non-linear movements such as rotation or a zigzag movement in the air, which can cause tracking with IoU to fail.

Algorithm 1 presents the pseudocode for the proposed identity association method for drone-captured inputs. $f_{metric}(\cdot)$ represents a linear assignment solver with *metric*. It also corresponds to the cost function $\mathcal{C}$ in Equation (1). It takes $D^{i-\tau}$, $T^{i-\tau}$, T^i as inputs, where τ denotes the frame gap of two consecutive detection processes. Then, it first assigns detections $D^{i-\tau}$ to tracks $T^{i-\tau}$ based on IoU such as usual association algorithms. The remaining tracks T_{remain} are categorized into two groups: T^{lost}_{remain} and T^{normal}_{remain}. T^{lost}_{remain} consists of the tracks that are being tracked in frame $i-\tau$ but are lost in the current frame i. On the other hand, T^{normal}_{remain} includes the tracks being tracked normally. For T^{normal}_{remain}, we use Euclidean distance as the metric for the linear assignment solver. This type of tracks typically indicates the drone's fast linear movement, which makes IoU nearly zero. For lost tracks, which may involve non-linear drone motion (e.g., rotation) and reappearing after occlusion, we use the color histogram metric. Finally, the unmatched detections are initialized as new tracks.

Algorithm 1 An Ensemble for Identity Association

procedure IDENTITY_ASSOCIATION($D^{i-\tau}$, $T^{i-\tau}$, T^{i})
  $T^{IoU}_{matched}, T_{remain}, D_{remain} \leftarrow f_{IoU}(D^{i-\tau}, T^{i-\tau})$
  for t_k in T_{remain} **do**
    if t_k is lost in T^i **then**
      Append t_k to T^{lost}_{remain}
    else
      Append t_k to T^{normal}_{remain}
    end if
  end for
  $T^{Euc}_{matched}, D_{remain} \leftarrow f_{Euc}(D_{remain}, T^{normal}_{remain})$
  $T^{Color}_{matched}, D_{remain} \leftarrow f_{Color}(D_{remain}, T^{lost}_{remain})$
  $T_{new} \leftarrow D_{remain}$ ▷ Initialize unmatched detections
  return $T^{i+1} \leftarrow \{T^{IoU}_{matched}, T^{Euc}_{matched}, T^{Color}_{matched}, T_{new}\}$
end procedure

The proposed identity association algorithm is implemented in C and provides fast yet accurate matching performance for drone-captured inputs.

4. Experiments

We evaluate the proposed real-time on-drone pedestrian tracker with real-world drone-captured videos in the DNA+Drone dataset [15]. The mission computer of our reference drone is either the NVIDIA Jetson TX2 (say, TX2), in which equips with a dual-core Denver2 CPU and 256-core Pascal GPU are integrated. We also evaluated on the NVIDIA Jetson AGX Xavier (say, Xavier), which is equipped with 8-core Camel ARM CPU and 512-core Volta GPU. As we will demonstrate, TX2 accomplishes real-time pedestrian tracking in 4K inputs. Consequently, contemporary platforms such as the NVIDIA Jetson AGX Orin series will exhibit higher efficiency while adhering to real-time constraints. Our reference drone is equipped with a with Sony A7m3 camera with a gimbal stabilization system.

Our implementation adopts YOLOv5s [36] as the detector and MOSSE [19] as the tracker. While there are various YOLO detector variants, such as YOLOv7 [37], the fundamental ideas presented in this paper remain consistent across different detector versions. The detector was fine-tuned using a part of the VisDrone [14] and DNA+Drone [15] datasets from the official pretrained model. The proposed identity association extends that in SORT [11]. The color histogram metric in identity association uses OpenCV's implementation of Bhattacharyya distance.

Note that in our experiments, we prioritize execution time over accuracy since our objective is to meet real-time constraints rather than achieving superior accuracy. It is crucial that the proposed tracker's accuracy lower bound matches that of SORT [11], as we have built upon it.

4.1. Efficiency Evaluation

In our initial evaluation, we concentrate on assessing the latency and update frequency of the proposed method. As we mentioned, our emphasis lies in prioritizing execution time over accuracy. Unlike many traditional tracking-by-detection algorithms that concentrate on accuracy metrics while overlooking system-level efficiency, our major objective is to attain real-time detection and tracking capabilities without compromising accuracy by the collaboration of detection and tracking. In the evaluation, we observed that the proposed method can successfully achieve the real-time performance.

Tables 1 and 2 show the breakdown of processing time on our mission computers. Using the TX2, the GPU-based detection phase takes 785.3 ms, resulting in 1.3 Hz of update frequency, approximately. The identity association step requires a total of 14.1 ms, but its update frequency is also limited to 1.3 Hz, as identities are updated only when there is a detection update. On the other hand, by leveraging our proposed parallel execution

approach, our tracking system achieves a 30.0 Hz update frequency, which satisfies real-time constraints. It is worth noting that updating at a rate of over 30 Hz is unnecessary, as it is bounded by the input video stream's frequency. If a higher update frequency is necessary, the proposed tracker can handle updates of up to 34.8 Hz.

Table 1. Breakdown of average processing time on NVIDIA Jetson TX2.

Module	Processor	Latency	Update Freq.
Detection	GPU	785.3 ms	1.3 Hz
ID-A (IoU)	CPU	4.1 ms	1.3 Hz
ID-A (Euc.)	CPU	0.9 ms	1.3 Hz
ID-A (Color)	CPU	9.1 ms	1.3 Hz
Tracking	CPU	28.7 ms	**30.0 Hz**

Table 2. Breakdown of average processing time on NVIDIA Jetson AGX Xavier.

Module	Processor	Latency	Update Freq.
Detection	GPU	145.7 ms	4.4 Hz
ID-A (IoU)	CPU	2.4 ms	4.4 Hz
ID-A (Euc.)	CPU	0.7 ms	4.4 Hz
ID-A (Color)	CPU	6.5 ms	4.4 Hz
Tracking	CPU	18.6 ms	**30.0 Hz**

With the Xavier, which has higher computational power than the TX2, a higher detection update frequency of 4.4 Hz can be achieved. Furthermore, the tracker can attain a maximum update frequency of 53.8 Hz, which can be useful when a camera with a higher frequency (e.g., 60 Hz) is employed.

It is worth noting that the proposed system does not consider video decoding time, and we assume the 4K image is stored in the main memory. In real drone scenarios, the captured image is delivered directly from the camera to the memory without compression.

As we mentioned in Section 3.2, dividing a large image into multiple patches and batching it can lead to a higher efficiency. However, smaller patch size is not necessarily translated as shorter execution time due to the fact that overlap at the boundary of patches is required so that an object at the boundary is not missed. The results shown in Figure 4 echo that. It shows that a patch size of 1280 × 1280 pixels is the best choice. Note that a larger patch size than 1600 × 1600 pixels and a smaller size than 320 × 320 just present worse latency. Regardless of the patch size, the processing time for slicing and stitching is negligible, taking approximately 3–5 ms in total. In contrast, detection takes a few hundred milliseconds.

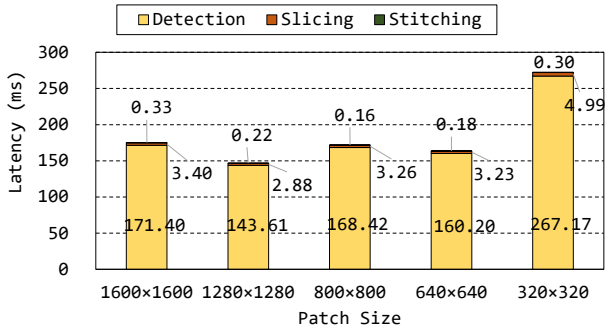

Figure 4. The latency of the detection phase on the GPU with respect to the patch size.

4.2. Accuracy Evaluation

While our focus remains on execution time, it is crucial to ensure that the approach does not compromise its inherent accuracy. In this section, we evaluate the proposed method with the existent classical approach, employing accuracy metrics. This comparative analysis demonstrates the generalizability of our proposed method.

In Table 3, we compare the performance with SORT [11] on Xavier. For this comparison, we utilize the MOT15 benchmark suite [38]. However, it is important to note that the MOT15 benchmark suite does not include 4K video sequences. Therefore, we only employ videos with a resolution of 1920×1080 pixels from the suite. Consequently, the update frequency of SORT in Table 3 is higher than the detection frequency denoted in Table 2, as the input size is smaller. The images in the suite are converted into a 24 FPS video. As a result, the maximum update frequency is 24.0 Hz in this evaluation, while the maximum update frequency in the previous experiments in Section 4.1 was 30.0 Hz. The proposed approach successfully achieved this maximum frequency of 24.0 Hz, with a latency of 14.0 ms.

Table 3. Evaluation of accuracy on MOT15 benchmark.

Method	HOTA	DetA	AssA	LocA	Update Freq.
SORT	10.81	10.45	12.24	66.70	9.1 Hz
Ours	18.38	18.78	18.71	71.48	24.0 Hz

It is crucial to emphasize that the detection performance significantly impacts the tracking performance. In comparison, we employ the same detector used in previous experiments, which is based on YOLOv5s. This detector is specifically trained to detect very tiny pedestrians in images captured from high altitudes. As a result, the absolute performance values are not optimal since the object size in the MOT15 benchmark, which comprises solely terrestrial videos, is comparatively larger. Nevertheless, it holds a notable meaning that the proposed method outperforms the traditional tracking algorithm. The comparison utilizes the HOTA metrics [39], a higher-order metric for evaluating multi-object tracking. The HOTA is a high-order metric that represents a combined result of three scores including detection accuracy (DetA), association accuracy (AssA), and localization accuracy (LocA). Overall, the proposed approach demonstrates superior performance compared to SORT across all metrics. The other kinds of conventional tracker can be compared relatively with the result of SORT.

Given that we utilize the same detector and similar tracking algorithms (both based on the Kalman filter) in the comparison, the significant difference between our method and SORT in terms of accuracy comes from the collaboration of detection and tracking, a concept we introduce. In particular, our tracking-assisted confidence boosting technique proposed in Section 3.2.3 plays a significant role. This technique substantially increases detection accuracy by recommending a mitigation in the confidence threshold for detection in situations where detected candidates align with active tracking objects. As a result, the gap between two approaches in detection accuracy is much higher than others, as shown in Table 3.

We count the number of rejected and recovered objects to test the efficacy of the proposed tracking-assisted confidence boosting technique. The result is illustrated in Figure 5. The thresholds of confidence score are set as 0.5 (θ_{conf}) and 0.3 (θ_{tacb}). Among the candidates that have low confidence score ($x_{conf} < \theta_{conf}$), 64% of them are rejected directly without boosting, as they have too low score ($x_{conf} < \theta_{tacb}$). The other 36% of candidates are recommended and compared to the live objects being tracked. It is 17% of candidates that are actually recovered by our boosting technique.

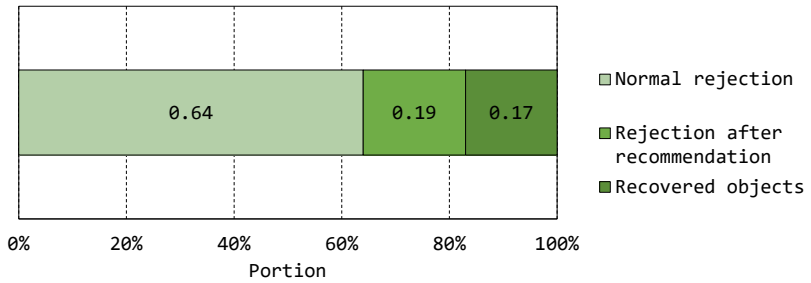

Figure 5. The portion of the recovered objects by tracking-assisted confidence boosting.

5. Conclusions

This paper presents a pedestrian tracker for drones that operates in real time and takes 4K images and videos captured at high altitudes as input. With the increasing importance of unmanned aerial vehicles (UAVs) in various fields, there is a growing need for drones to process tasks in real time to make immediate decisions. This makes it crucial to develop efficient and effective systems that can operate on drones' limited computational resources.

We propose several key ideas, including intermittent detection and parallel execution, and identity an association method for drone-captured inputs to address the challenges of on-drone processing. These ideas, along with other optimizations, enable our pedestrian tracker to achieve real-time processing on 4K input streams using an NVIDIA Jetson TX2.

Author Contributions: Conceptualization, M.L.; methodology, M.L. and C.O.; software, C.O. and M.L.; validation, C.O.; formal analysis, C.O.; resources, M.L. and C.L.; writing—original draft preparation, C.O.; writing—review and editing, C.O.; supervision, C.O.; project administration, M.L. and C.L.; funding acquisition, C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the DNA+Drone Technology Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (No. NRF-2020M3C1C2A01080819).

Data Availability Statement: The DNA+Drone Dataset is available at: https://nanum.etri.re.kr/dnaplusdrone (accessed on 3 October 2023).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
SOT	Single Object Tracking
MOT	Multiple Object Tracking
IoU	Intersection Over Union
TACB	Tracker-Assisted Confidence Boosting
CPU	Central Processing Unit
GPU	Graphic Processing Unit
FC	Flight Controller
MC	Mission Computer
FPS	Frames Per Second
SoC	System-on-Chip
CNN	Convolutional Neural Network
PSR	Peak-to-Sidelobe Ratio
KCF	Kernelized Correlation Filters
CSRT	Discriminative Correlation Filter with Channel and Spatial Reliability Tracker
MOSSE	Minimum Output Sum of Squared Error

References

1. Puttock, A.; Cunliffe, A.; Anderson, K.; Brazier, R.E. Aerial Photography Collected with a Multirotor Drone Reveals Impact of Eurasian Beaver Reintroduction on Ecosystem Structure. *J. Unmanned Veh. Syst.* **2015**, *3*, 123–130. [CrossRef]
2. Ding, G.; Wu, Q.; Zhang, L.; Lin, Y.; Tsiftsis, T.A.; Yao, Y.D. An Amateur Drone Surveillance System Based on the Cognitive Internet of Things. *IEEE Commun. Mag.* **2018**, *56*, 29–35. [CrossRef]
3. Xu, C.; Zhang, K.; Jiang, Y.; Niu, S.; Yang, T.; Song, H. Communication aware UAV swarm surveillance based on hierarchical architecture. *Drones* **2021**, *5*, 33. [CrossRef]
4. Tariq, R.; Rahim, M.; Aslam, N.; Bawany, N.; Faseeha, U. Dronaid: A Smart Human Detection Drone for Rescue. In Proceedings of the 2018 15th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT (HONET-ICT), Islamabad, Pakistan, 8–10 October 2018; pp. 33–37.
5. Schedl, D.C.; Kurmi, I.; Bimber, O. An Autonomous Drone for Search and Rescue in Forests using Airborne Optical Sectioning. *Sci. Robot.* **2021**, *6*, eabg1188. [CrossRef] [PubMed]
6. Besada, J.A.; Bergesio, L.; Campaña, I.; Vaquero-Melchor, D.; López-Araquistain, J.; Bernardos, A.M.; Casar, J.R. Drone Mission Definition and Implementation for Automated Infrastructure Inspection using Airborne Sensors. *Sensors* **2018**, *18*, 1170. [CrossRef] [PubMed]
7. Balamuralidhar, N.; Tilon, S.; Nex, F. MultEYE: Monitoring system for real-time vehicle detection, tracking and speed estimation from UAV imagery on edge-computing platforms. *Remote Sens.* **2021**, *13*, 573. [CrossRef]
8. Rančić, K.; Blagojević, B.; Bezdan, A.; Ivošević, B.; Tubić, B.; Vranešević, M.; Pejak, B.; Crnojević, V.; Marko, O. Animal Detection and Counting from UAV Images Using Convolutional Neural Networks. *Drones* **2023**, *7*, 179. [CrossRef]
9. Zhang, Y.; Wang, C.; Wang, X.; Zeng, W.; Liu, W. FairMOT: On the Fairness of Detection and Re-identification in Multiple Object Tracking. *Int. J. Comput. Vis.* **2021**, *129*, 3069–3087. [CrossRef]
10. Liu, Q.; Chu, Q.; Liu, B.; Yu, N. GSM: Graph Similarity Model for Multi-Object Tracking. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20), Yokohama, Japan, 11–17 July 2020; pp. 530–536.
11. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple Online and Realtime Tracking. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
12. Wojke, N.; Bewley, A.; Paulus, D. Simple Online and Realtime Tracking with a Deep Association Metric. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3645–3649.
13. Zhang, Z.; He, Y.; Guo, H.; He, J.; Yan, L.; Li, X. Towards Robust Visual Tracking for Unmanned Aerial Vehicle with Spatial Attention Aberration Repressed Correlation Filters. *Drones* **2023**, *7*, 401. [CrossRef]
14. Fan, H.; Du, D.; Wen, L.; Zhu, P.; Hu, Q.; Ling, H.; Shah, M.; Pan, J.; Schumann, A.; Dong, B.; et al. VisDrone-MOT2020: The Vision Meets Drone Multiple Object Tracking Challenge Results. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Glasgow, UK, 23–28 August 2020; pp. 713–727.
15. Lim, Y.; Kim, Y.; Lim, C. DNA+Drone: Drone Service Platform Converging Bigdata, 5G Networks, and AI, which are Korean ICT Strengths. In Proceedings of the US-Korea Conference on Science, Technology, and Entrepreneurship (UKC), Arlington, TX, USA, 17–20 August 2022; p. 265.
16. Hong, Y.; Kim, S.; Kim, Y.; Cha, J. Quadrotor Path Planning using A* Search Algorithm and Minimum Snap Trajectory Generation. *ETRI J.* **2021**, *43*, 1013–1023. [CrossRef]
17. Canovas, B.; Nègre, A.; Rombaut, M. Onboard Dynamic RGB-D Simultaneous Localization and Mapping for Mobile Robot Navigation. *ETRI J.* **2021**, *43*, 617–629. [CrossRef]
18. Hong, T.; Liang, H.; Yang, Q.; Fang, L.; Kadoch, M.; Cheriet, M. A real-time tracking algorithm for multi-target UAV based on deep learning. *Remote Sens.* **2022**, *15*, 2. [CrossRef]
19. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual Object Tracking using Adaptive Correlation Filters. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, PR, USA, 17–19 June 2010; pp. 2544–2550.
20. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-speed Tracking with Kernelized Correlation Filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *37*, 583–596. [CrossRef] [PubMed]
21. Lukezic, A.; Vojir, T.; ˇCehovin Zajc, L.; Matas, J.; Kristan, M. Discriminative Correlation Filter with Channel and Spatial Reliability. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6309–6318.
22. Wu, L.; Wang, Y.; Shao, L.; Wang, M. 3-D PersonVLAD: Learning Deep Global Representations for Video-based Person Reidentification. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3347–3359. [CrossRef] [PubMed]
23. Sekh, A.A.; Dogra, D.P.; Choi, H.; Chae, S.; Kim, I.J. Person Re-identification in Videos by Analyzing Spatio-Temporal Tubes. *Multimed. Tools Appl.* **2020**, *79*, 24537–24551. [CrossRef]
24. Hou, R.; Ma, B.; Chang, H.; Gu, X.; Shan, S.; Chen, X. VRSTC: Occlusion-free Video Person Re-identification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7183–7192.
25. Wang, G.; Lai, J.; Huang, P.; Xie, X. Spatial-temporal Person Re-identification. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 8933–8940.
26. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.

27. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks. In Proceedings of the Advances in Neural Information Processing Systems 28 (NIPS 2015), Montreal, QC, Canada, 7–12 December 2015; Volume 28.
28. Zhu, X.; Lyu, S.; Wang, X.; Zhao, Q. TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11–17 October 2021; pp. 2778–2788.
29. Voigtlaender, P.; Krause, M.; Osep, A.; Luiten, J.; Sekar, B.B.G.; Geiger, A.; Leibe, B. MOTS: Multi-object Tracking and Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7942–7951.
30. Xu, J.; Cao, Y.; Zhang, Z.; Hu, H. Spatial-temporal Relation Networks for Multi-object Tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3988–3998.
31. Sadeghian, A.; Alahi, A.; Savarese, S. Tracking the Untrackable: Learning to Track Multiple Cues with Long-term Dependencies. In Proceedings of the IEEE international Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 300–311.
32. Xiao, C.; Cao, Q.; Zhong, Y.; Lan, L.; Zhang, X.; Cai, H.; Luo, Z. Enhancing Online UAV Multi-Object Tracking with Temporal Context and Spatial Topological Relationships. *Drones* **2023**, *7*, 389. [CrossRef]
33. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is All You Need. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
34. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision Meets Robotics: The KITTI Dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
35. Zhu, P.; Wen, L.; Du, D.; Bian, X.; Ling, H.; Hu, Q.; Nie, Q.; Cheng, H.; Liu, C.; Liu, X.; et al. VisDrone-DET2018: The Vision Meets Drone Object Detection in Image Challenge Results. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018.
36. Jocher, G. YOLOv5 by Ultralytics. 2020. Available online: https://github.com/ultralytics/yolov5 (accessed on 21 August 2023).
37. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
38. Leal-Taixé, L.; Milan, A.; Reid, I.; Roth, S.; Schindler, K. MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking. *arXiv* **2015**, arXiv:1504.01942.
39. Luiten, J.; Osep, A.; Dendorfer, P.; Torr, P.; Geiger, A.; Leal-Taixé, L.; Leibe, B. HOTA: A Higher Order Metric for Evaluating Multi-Object Tracking. *Int. J. Comput. Vis.* **2020**, *129* , 548–578. [CrossRef] [PubMed]