SkyNet: Multi-Drone Cooperation for Real-Time Person Identification and Localization

Junkun Peng*†, Qing Li†¶, Yuanzheng Tan‡, Dan Zhao†, Zhenhui Yuan§,
Jinhua Chen‡, Hanling Wang*† and Yong Jiang*†
*Tsinghua Shenzhen International Graduate School, Shenzhen, China
†Peng Cheng Laboratory, Shenzhen, China
‡Sun Yat-Sen University, Shenzhen, China
§Northumbria University, Newcastle upon Tyne, United Kingdom

IEEE INFOCOM 2023 - IEEE Conference on Computer Communications | 979-8-3503-3414-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/INFOCOM53939.2023.10228978

***Abstract*—Aerial images from drones have been used to detect and track suspects in the crowd for the public safety purpose. However, using a single drone for human identification and localization faces many challenges including low accuracy and long latency, due to poor visibility, varying field of views (FoVs), and limited on-board computing resources. In this paper, we propose SkyNet, a multi-drone cooperative system for accurate and real-time human identification and localization. SkyNet computes the 3D position of a person by cross searching from multiple views. To achieve high accuracy in identification, SkyNet fuses aerial images of multiple drones according to their legibility. Moreover, by predicting the estimated finishing time of tasks, SkyNet schedules and balances workloads among edge devices and the cloud server to minimize processing latency. We implement and deploy SkyNet in real life, and evaluate the performance of identification and localization with 20 human participants. The results show that SkyNet can locate people with an average error within 0.18m on a square of 554m^2. The identification accuracy is 91.36%. The localization and identification process is completed within 0.84s.**

***Index Terms*—multi-drone, person identification, localization, task distribution, information fusion.**

I. Introduction

Human tracking and identification technology [1], [2] has been widely used to improve public safety [3]–[5]. Existing solutions mainly rely on images captured by fixed-position cameras [6], [7], which have limited field-of-views (FoVs) and are inefficient for tracking moving objects. Benefiting from the wide FoVs and high mobility, drone-based human identification and tracking solutions can be applied in many application scenarios, e.g., military actions and security services [8].

Recently, DNN-based face identification solutions have achieved high accuracy, but it is on the premise of sufficient amounts of face pixels [9], [10]. However, a single drone often suffers from limited face pixels due to its high-flying height and varying drone-person angles, which is revealed in the motivational studies in Section II. DNN-based face identification technologies also consume massive on-board computing resources and bring high latency to the system, which is not ideal for real-time identification.

To track a person in the crowd, most state-of-the-art localization technologies require RGB-D cameras or LiDAR on drones, which are 10 times more expensive than conventional 2D cameras. Moreover, in outdoor and long-distance scenarios, the positioning accuracy and point cloud density of RGB-D cameras and LiDAR drop dramatically, leading to poor localization accuracy and a small working area [11], [12].

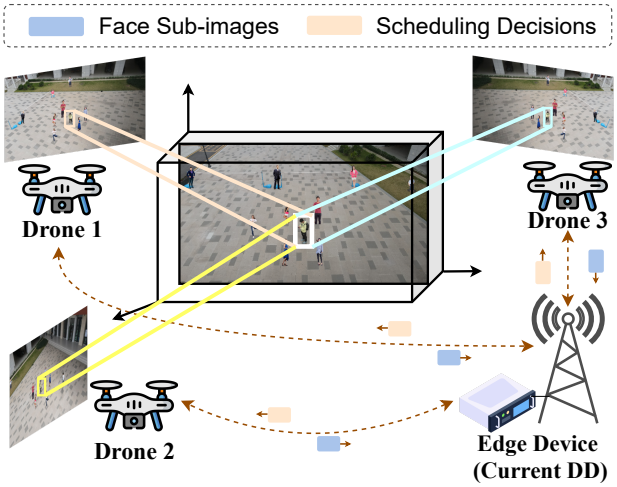


Fig. 1: SkyNet uses multiple drones and edge devices to identify and locate the target person.

In this paper, we design SkyNet, a multi-drone cooperative framework to achieve accurate and real-time identification and localization. SkyNet provides a universal framework that can be applied for person localization/identification. Taking the former as an example, as shown in Figure 1, at a shooting instant, each drone takes a picture of the crowd and uses on-board person/face detection models to detect face sub-images, which are sent to the designated device (DD) selected from edges and the cloud server. Then the DD computes the 3D location of each face and generates identification results by jointly exploiting the images from multiple views.

To achieve SkyNet's high-level design goals, several challenges need to be addressed. i) How to find a person's 3D position and align his/her face sub-images from multiple views when there are only 2D images of him/her in different views? In fact, forming a unified perception of the scene is a common challenge faced by multi-drone/robot cooperation. ii) How to take advantage of the face information offered by multi-view images for much more accurate identification? iii) With the

¶ Corresponding author: Qing Li (liq@pcl.ac.cn)

system taking pictures continuously, how to select the suitable DD to ensure fast processing?

To address these challenges, in SkyNet, we propose an innovative *Multi-view Cross Search Algorithm* to efficiently find the 3D real-world location of a face and align his/her multiple 2D sub-images. To improve identification accuracy, we propose a novel *Fusion Weight Network (FWN)*, which generates fusion weight factors for a person's face sub-images from different FoVs, based on which these sub-images are fused for inference. Moreover, we propose a *Dynamic Task Scheduling Algorithm* to balance workloads over consecutive shooting instants and reduce processing latency.

The key contributions of this paper are as follows.

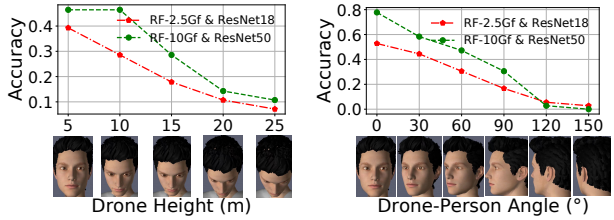
- We design a multi-drone cooperative framework to achieve real-time identification and localization.
- We propose an algorithm for accurately locating a person in 3D real world using only conventional 2D cameras and aligning face sub-images of one person from different drone views. The algorithm allows multiple drones to form a unified perception of the real-world scene.
- We design a novel fusion identification pipeline, which takes advantage of images from different FoVs by fusing them with weights that reflect legibility. The pipeline also reduces processing latency by parallel computing.
- SkyNet reduces the latency of task processing to achieve real-time execution through the cooperation of heterogeneous devices and dynamic task scheduling.
- We implement and deploy SkyNet in real life on four drones, three edge devices, and a cloud server. We not only test SkyNet with public datasets about drone-based face identification but also conduct real-world experiments with 20 human participants and obtain their consent. The evaluation results show that SkyNet achieves 91.36% accuracy and the real-time latency of localization and identification (within 0.84s).

II. Motivational Studies

In this section, we comprehensively analyze the impacts of the drone view on human identification accuracy and computational latency using a single drone.

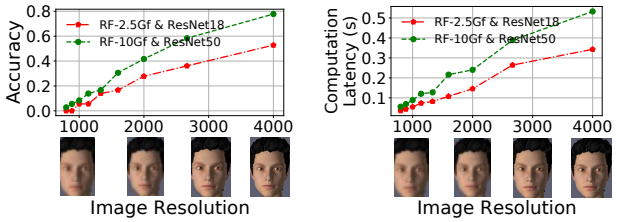
We use the NVIDIA Jetson Xavier NX [13] as the edge device to process the capturing images of the drone. The identification pipeline consists of RetinaFace [14] for face detection, and ResNet-based ArcFace [1] for face identification. We use two model configurations, RetinaFace-2.5Gf & ResNet18 and RetinaFace-10Gf & ResNet50, corresponding to the light model configuration and the heavy model configuration, respectively. We collect 567 drone images with the DJI Mavic [15] at various distances, heights, drone-person angles, and resolutions. By using these aerial images from the drone, we mainly analyze the impacts of flight height, drone-person angle, and image resolution on human identification accuracy and computational latency.

The high-flying height offers a broad view scope but degrades the face identification accuracy. Figure 2a presents



(a) Impacts of drone height on identification accuracy.

(b) Impacts of drone-person angle on identification accuracy.



(c) Impacts of image resolution on identification accuracy.

(d) Impacts of image resolution on computational latency.

Fig. 2: Identification service provided by a single drone. Animated faces are used in the figures instead of real person faces only for the double-blind review purpose.

the accuracy achieved at different heights. The horizontal distance between the drone and the target person to be identified is fixed at 20 meters, and the images used are in 4K resolution. We down sample these images according to the needs of the experiment. As the drone flies higher, the identification accuracy decreases significantly in both model configurations. The accuracy is even less than 10% at the height of 25m since fewer pixels are available for the target person.

The large drone-person angle results in low identification accuracy. Figure 2b illustrates the accuracy achieved at different drone-person angles (0°: the drone exactly faces the frontal face of the person, and 180°: the drone exactly faces the back of the head of the person). In this experiment, the height is 5 meters, the horizontal distance is 10 meters and the resolution is 4K. The result shows that as the drone-person angle increases, the accuracy drops significantly and nearly reaches zero at around 150°.

The high image resolution boosts the accuracy but incurs the high computational latency. Figure 2c and Figure 2d show the identification accuracy and computational latency under 9 photo resolution settings, i.e., 4000×3000, 2666×2000, 2000×1000, 1600×1200, 1333×1000, 1142×857, 1000×750, 888×400, 800×600. For both model configurations, the better resolution leads to higher accuracy, while causing significantly higher computational latency.

According to the above results, the higher the flight height, the farther the distance between the drone and the target person, and the steeper the drone-human angle, resulting in fewer effective pixels available, which in turn leads to lower accuracy. Furthermore, higher image resolutions lead to larger transmission and inference latency, but lower resolutions reduce the accuracy of face recognition. To this end, using multiple drones instead of a single drone could be an alternative solution for human identification.

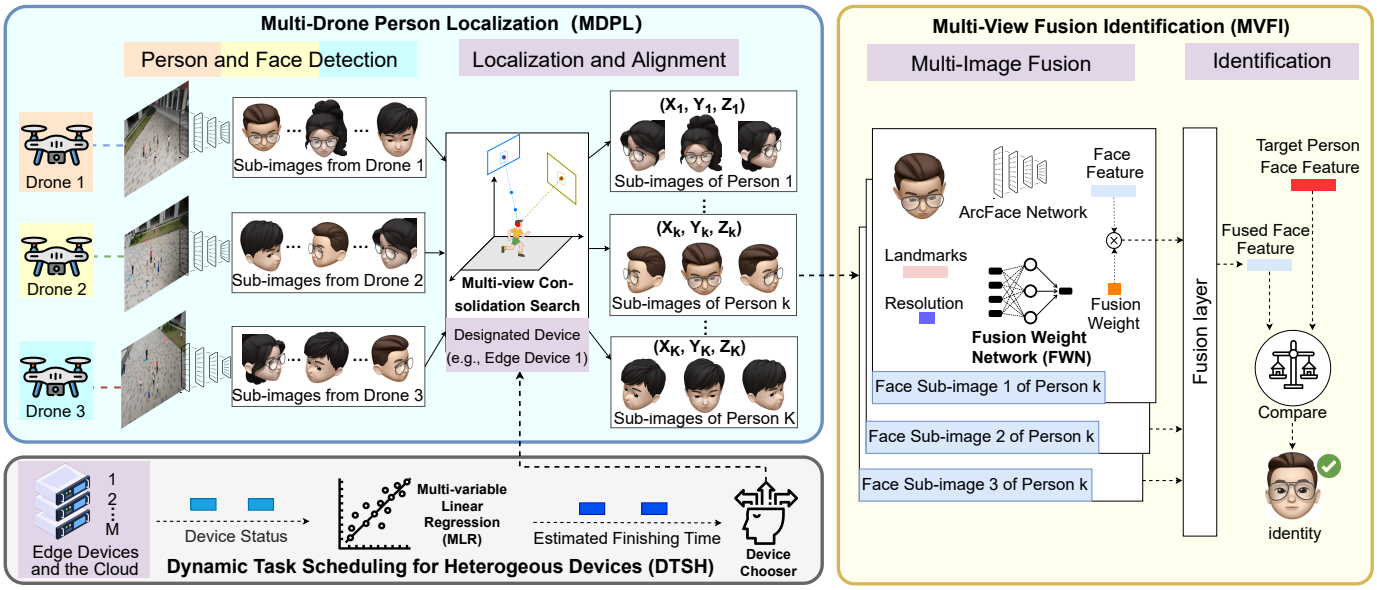


Fig. 3: Operational Flow of SkyNet in a nutshell.

III. SkyNet: Overview

We build a multi-drone cooperative framework to locate and identify persons in a crowded scene, which consists of:

i) a group of drones with computing capabilities that can capture images from different angles, run lightweight models, and offload computation tasks to other devices.

ii) a group of interconnected edge devices, closer to drones, can run models but with finite computing power. In SkyNet, a stable edge device is selected as the home edge, which collects status information from other edges and schedule tasks.

iii) a cloud server, farther from drones and edge devices, but with sufficient computing power. If the workload of all edge devices is too heavy, the computation can be offloaded to it.

The operational flow of SkyNet is illustrated in Figure 3. In a crowded scene, SkyNet locates and identifies the human target using the Multi-Drone Person Localization (MDPL) and the Multi-View Fusion Identification (MVFI) modules. The Dynamic Task Scheduling for Heterogeneous Devices (DTSH) module is designed to schedule tasks among edge devices and the cloud server to reduce latency and balance workload.

i) **MDPL**. On each drone, aerial images of the crowd are first processed on-board by a person detection model and a face detection model. This extracts face sub-images with their bounding box locations and face landmarks and sends them to the DD selected via the DTSH module. Then SkyNet runs the MDPL module on the DD to locate each person's real-world location through a series of coordinate matrix transformations and align his/her sub-images from multiple views.

ii) **MVFI**. After aligning the multi-view sub-images of a person, SkyNet runs the MVFI module on the DD. The MVFI first extracts face features from all face sub-images captured at different angles. Second, to comprehensively utilize features from these different angles, the FWN assigns a fusion weight to each sub-image to reflect its legibility. Then, the fusion layer generates the fused face feature by combining face features from different views using fusion weights. The fused face feature contains more details than those extracted from a single image. Finally, the fused face feature is compared with the facial feature of the target person by calculating their feature distance. A person whose feature distance is within the threshold is considered the target person to be found.

iii) **DTSH**. We define the entire operation flow of localization and recognition as a *task*. At a shooting instant, each drone takes a photo of the crowd. The detection part of one task, i.e., face sub-image extraction in MDPL, is executed in parallel on each drone. The remaining parts of one task need to use images from all views simultaneously and thus can only be done on one device, i.e., the DD. Therefore, the remaining parts of one task are called the DD-side task, including the localization sub-task, the fusion sub-task, and the identification sub-task. In order to balance the workload of edge devices and ultimately reduce the processing latency, selecting a suitable DD for one DD-side task is the key. For this purpose, the home edge first predicts the Estimated Finishing Time (EFT) required by each edge device if it handles this DD-side task. Then, it selects the device with the shortest EFT as the DD for this task. If all the EFTs of edge devices are higher than the shooting interval (e.g., 1s), it selects the cloud server as the DD of this task. Finally, scheduling decisions are sent to all drones, which offload data to the DD.

IV. Multi-Drone Person Localization

In this section, we present the process of using multiple drones for person localization. First, the face sub-image extraction model is designed. Then, a multi-view cross-localization strategy is proposed to find the 3D position of each person and align his/her sub-images from different views.

A. Face Sub-images Extraction

The first step of the MDPL module is to extract face sub-images, their bounding box locations, and face landmarks on

each drone. Because drones are far from the crowd, some faces are too small to be recognized in images. We first use person detection to find the full body of each person on images. Then, we use face detection to detect each person's face.

Considering the limited computational and power resources of drones, we choose YOLOX-Tiny detector [16] as the person detection network and RetinaFace detector [14] as the face detection network because of their accurate detection rate [17], [18] and fast processing speed [18], [19]. The person detection network uses the DNN to generate the bounding box position of each person in the original image, and the face detection generates the bounding box position and facial landmark position of each face. Next, on each drone, we can get the bounding box location of each face and its face landmarks, and then extract the face sub-images. Each drone then sends this information to the DD, which is selected by the DTSH module described in section VI for further processing, namely localization, alignment, feature fusion, and identification.

B. Multi-View Cross Localization and Alignment

Face sub-images and bounding box locations from multiple views are aggregated in the DD, but it is unclear which sub-images from multiple drones belong to the same person. In this subsection, we propose a *multi-view cross search algorithm* to determine a person's 3D real-world location and align face sub-images of the same person from different drones by the location. First, we randomly select a drone D_i and use its view as the *primary view*. Denote the center point of the face in the pixel coordinates of the drone D_i as $P_i = [x_i, y_i]^T$.

In order to find the 3D real-world position of this face, denoted as $P^W = [x^W, y^W, z^W]^T$, we first establish the transformation relationship between P_i and P^W, as follows:

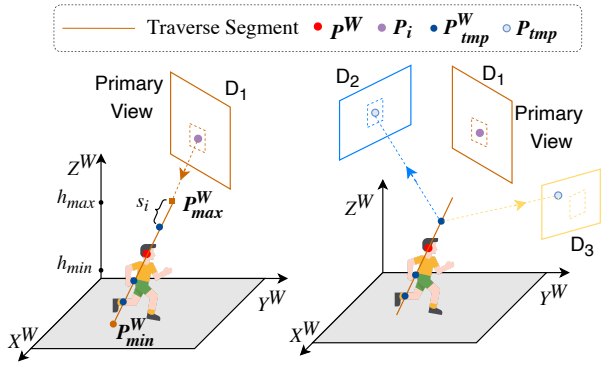
$$z_i^C \begin{bmatrix} P_i \\ 1 \end{bmatrix} = \begin{bmatrix} C_i & \overrightarrow{0} \end{bmatrix} \begin{bmatrix} R_i & T_i \\ \overrightarrow{0} & 1 \end{bmatrix} \begin{bmatrix} P^W \\ 1 \end{bmatrix}, \quad (1)$$

where $C_i \in \mathbb{R}^{3\times3}$ denotes the *camera internal matrix* of the drone D_i, which can establish the mapping relationship between the image pixel coordinates and the camera coordinates for D_i. $R_i \in \mathbb{R}^{3\times3}$ and $T_i \in \mathbb{R}^{3\times1}$ represent the *rotation matrix* and *translation matrix* of the drone D_i, which could establish the mapping relationship between the camera coordinates and the world coordinates for D_i. R_i and T_i can be obtained by the Perspective-N-Points (PNP) positioning[1] [20] and C_i can be obtained by the camera calibration technique [21]. Following (1), P_W is given as:

$$P^W = R_i^{-1}\left[z_i^C C_i^{-1} P_i - T_i\right]. \quad (2)$$

Due to the absence of depth information z_i^C, P^W is still a variable depending on z_i^C. Hence, we find the final piece of the puzzle by exploiting information provided by the views of other drones.

By varying z_i^C in (2), P^W forms a line in 3D space, called the inverse line, denoted as l. Without any other additional information about z_i^C, we traverse the points on l using an adaptive stride. We consider the highest and lowest possible heights of a person in the real world, denoted as $h_{\max}$ and $h_{\min}$, and get the corresponding real-world locations $P^W_{\max}$ and $P^W_{\min}$ by these two heights. The z_i^C corresponds to $P^W_{\max}$ and $P^W_{\min}$, denoted as $z_{\max}$ and $z_{\min}$, respectively, can be calculated according to (1). By restricting the inverse line l with $z_{\max}$ and $z_{\min}$, the search space l is further narrowed down to a line segment, named the *traverse segment*. The traverse stride of z_i^C is set as $s_i = \frac{z_{\max}-z_{\min}}{(h_{\max}-h_{\min})/s^W}$, where the s^W is the fixed real-world height stride (e.g., 10 cm).



(a) Space inverse transformation. (b) Cross check for a traverse point.

Fig. 4: Multi-View Cross Localization.

As shown in Figure 4, for each traverse point P^W_{tmp}, we conduct a cross-check by projecting it onto all other drones' views using (1). On each view, if the projected point falls into a face bounding box, we assert P^W_{tmp} is the true point P^W, i.e., the 3D real-world position of the face. Note that in this process of cross-checking, the corresponding sub-images of the person in all the different views are found, effectively accomplishing the alignment. The pseudo code of the algorithm is shown in Algorithm 1. By executing Algorithm 1, we can match all the faces in the primary view with the faces in other drones' views. If there are unmatched faces in other drones' views, it indicates that occlusion or missing detection has occurred in the primary view. We change the primary view to the drone containing unmatched faces and perform Algorithm 1 on unmatched faces, in order to ensure that all faces in all views are aligned. The more drones involved in the collaboration, the higher the accuracy of the match.

V. Multi-View Fusion Identification

In this section, we detail our proposed multi-view fusion recognition pipeline and algorithm, as well as a parallel computing strategy aimed at speeding up processing.

A. Multi-view Face Feature Fusion

For each face sub-image, we use ArcFace [1] to extract a 512-dimensional feature vector. In SkyNet, we fuse face features from different angles according to the weights generated by FWN to achieve more accurate recognition. Because if only the vector with the smallest distance from the target face

[1]As SkyNet is designed to locate a person in a known space, it is feasible to get known calibration points.

Algorithm 1 Multi-view Cross Search Algorithm

Input: i, $h_{\max}$, $h_{\min}$, N, s^W, P_i, for each drone:[R, T, C]

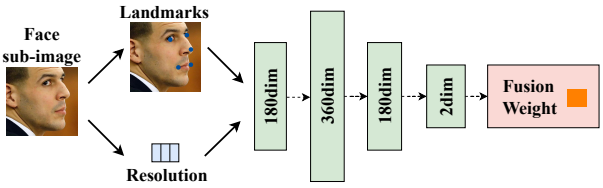
$l \leftarrow getInverseLine(P_i)$
$z_{\max} \leftarrow$ set Z^W of P^W in (2) to $h_{\max}$
$z_{\min} \leftarrow$ set Z^W of P^W in (2) to $h_{\min}$
$z_i^C = z_{\min}$, $s_i = \frac{z_{\max} - z_{\min}}{(h_{\max} - h_{\min})/s^W}$
while $z_i^C \leq z_{\max}$ **do**
  $P_{tmp}^W \leftarrow pixel2realworld(z_i^C, P_i)$
  $result$ = true
  for k in [1,N]\{i} **do**
    $P_k = realworld2pixel_{\text{view k}}(P_{tmp}^W)$
    if !$pointInBoundingBox(P_k)$ **then**
      $result$ = false
    end if
  end for
  if $result$ is true **then**
    $P^W \leftarrow P_{tmp}^W$
    break;
  end if
  $z_i^C \leftarrow z_i^C + s_i$;
end while

feature is selected for recognition, the information of other angle feature vectors will be wasted.

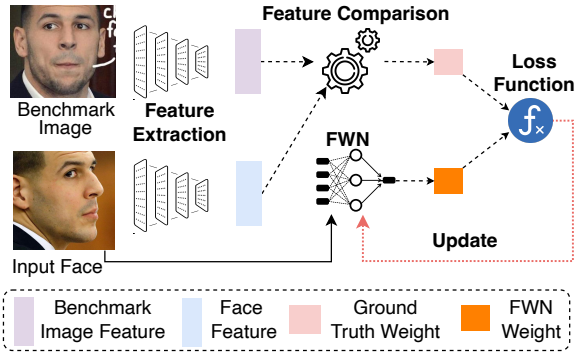
1) Fusion Weight Network: As demonstrated in Section II, face angle and resolution can seriously affect the accuracy of face recognition. To establish a reasonable fusion method for images of different angles and qualities, we design the FWN. As shown in Figure 5a, for each face image, the FWN takes its resolution and landmarks of face features as input and outputs the fusion weight of the image. This is achieved by networks capturing the hidden relationships between landmark vectors, resolution, and image legibility. Intuitively, face images that are easier to identify should be given higher fusion weights. To this end, *weight loss* for training FWN is defined as follows:

$$L_{weight} = \frac{1}{N}\sum_{1}^{N}\left(y - \frac{d}{||\phi(Face) - \phi(Face_{bm})||^2}\right), \quad (3)$$

where N is the number of samples in a training batch, y denotes the fusion weight of an input image generated by the FWN, d denotes the dimension of the feature vector, $Face$ and $Face_{bm}$ are the input image and the benchmark image (i.e., a frontal view image with sufficient effective pixel) of the person, respectively, and $\phi(\cdot)$ denotes the d-dimensional face feature vector extracted by face feature extraction. $\frac{d}{||\phi(Face) - \phi(Face_{bm})||^2}$ is also called ground truth weight. By minimizing the weight loss L_{weight}, an effective FWN can be trained to output fusion weights that can accurately reflect the legibility of each image, as shown in Figure 5b. We train the FWN on the CFP dataset [22]. Using the trained FWN, we can generate the corresponding fusion weights for the same person's face sub-images from different views, which are then normalized and fed into the fusion layer.



(a) Fusion Weight Network architecture.



(b) Training process of Fusion Weight Network.

Fig. 5: Fusion Weight Network

2) Fusion Layer and Identification Result: For each person, the MVFI first uses the fusion layer to fuse his/her face sub-image features from different views to obtain a fusion feature, which is calculated by the weighted sum of all-view features. The weights in the weighted sum are given by the FWN. Then the fusion layer calculates the distance between it and the target person's face feature. After getting the feature distances between each person and the target person, the minimum feature distance is compared with the feature distance threshold, and if the minimum feature distance is lower than the threshold, the corresponding person is identified as the target person.

B. Parallel Heterogeneous Computing

To guarantee real-time performance, we design a Parallel Heterogeneous Computing (PHC) strategy. The PHC sets up the thread pool to prepare for upcoming tasks. When face sub-images of people from multiple views are passed to the PHC, each sub-image is submitted to the thread pool as a job. The thread pool creates a thread for each job to realize multi-view parallel recognition. To further reduce the pipeline latency, we employ heterogeneous computation within each thread through CPU-GPU collaboration. Specifically, the FWN is executed by the CPU, while the relatively heavy face feature extraction model is executed by the GPU to fully utilize the overall computing power of devices.

VI. Dynamic Task Scheduling for Heterogeneous Devices

The scheduling of tasks is executed by the home edge. When the home edge schedules a task, it selects the DD from all edges and the cloud server to complete the DD-side task. To predict the EFT, the following information related to the states of devices needs to be considered: 1) the length of the

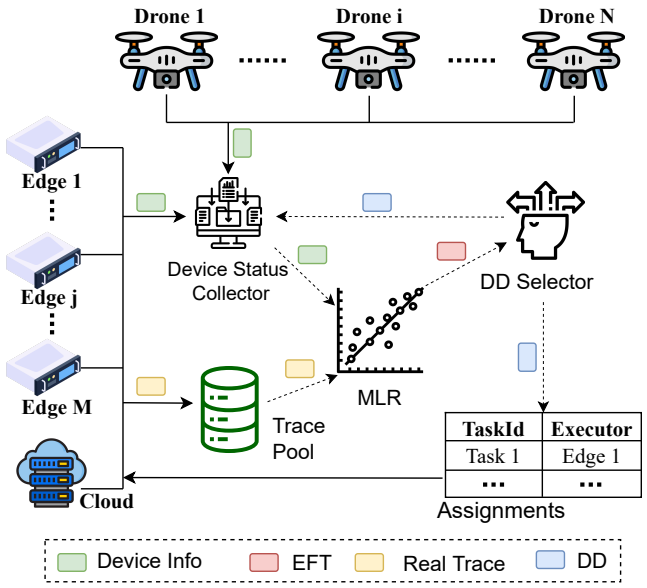


Fig. 6: Dynamic Task Scheduling for Heterogeneous Devices

current unfinished task queue for each candidate device; 2) the computing power of each candidate device, such as CPU and GPU resources; 3) the transmission latency between each candidate device and drones. To this end, the home edge needs to be up to date on all these three aspects of devices.

To avoid frequently obtaining information from all devices, the DTSH sets a scheduling period (e.g., 1 minute), which spans a series of tasks, denoted as $Queue_a$, which can be flexibly configured according to the system needs. The scheduling process in one scheduling period includes the following three steps.

1) Synchronization. At the beginning of each scheduling period, the home edge obtains the latest device state information from all edge devices, including queue lengths, available GPU/CPU resources, and current transmission latency.

2) Scheduling. The scheduling decision of one scheduling period made by the home edge includes selecting the suitable DD for each task within the period. When a device is already assigned multiple tasks, its queue length becomes longer, so it is less likely to be selected again. The home edge selects DDs for tasks one by one in chronological order of tasks because there is a temporal dependency between tasks.

3) Schedule decision dissemination. After all tasks are scheduled, the home edge sends the scheduling decision, i.e., the sequence of selected DDs, to all drones, edge devices, and the cloud server.

In the following, we explain how to select the DD for a task within a scheduling period. Consider N drones $\mathbb{D} = \{D_1, \cdots, D_i, \cdots, D_N\}$, M edge devices $\mathbb{E} = \{E_1, \cdots, E_j, \cdots, E_M\}$ and a cloud server, denoted as E_0. Suppose the current unfinished tasks queue of E_j is $Queue_j$, and the CPU and GPU computing powers of device E_j are CPU_j and GPU_j (unit: TOPS), respectively. The transmission latency of E_j is calculated as the maximum transmission delay between it and all drones $Delay_j = \max_{i \in [1,N]} Delay_{j,i}$, where $Delay_{j,i}$ is the transmission delay between E_j and D_i.

For a task, the home edge selects the device with the smallest EFT as the task's DD. If the EFTs of all edge devices for one task are higher than the shooting interval (e.g., 1s), indicating all of them are busy, the cloud server is selected as the task's DD. The estimated finishing time EFT_j of device E_j can be predicted by an adaptive Multi-variable Linear Regression (MLR) model:

$$\begin{aligned} EFT_j =& MLR(Delay_j, ||Queue_j||, CPU_j, GPU_j) \\ =& \theta_0 + \theta_1 Delay_j + \theta_2 Queue_j + \theta_3 GPU_j + \theta_4 CPU_j. \end{aligned} \tag{4}$$

To train the MLR model (4), a trace pool is built to store the real historical traces. Each trace is a pair of data that records the MLR input vector $(Delay_j, Queue_j, CPU_j, GPU_j)$ and the real finishing time (ground truth). The MLR is updated through training with the real traces in the trace pool in each updating period (e.g. 2 minutes).

The training purpose of MLR is to find out a parameter set $\hat{\boldsymbol{\theta}} = (\hat{\theta}_0, \hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3, \hat{\theta}_4)$, which is as close to the real parameter set $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2, \theta_3, \theta_4)$ as possible. Because the real $\boldsymbol{\theta}$ is unknown, we cannot directly compare the two. To describe how close $\hat{\boldsymbol{\theta}}$ is to $\boldsymbol{\theta}$, we introduce a loss function:

$$Loss = \frac{1}{K} \sum_{j=1}^{K} (EFT_j - \tilde{EFT}_j), \tag{5}$$

where K is the number of traces, EFT_j is the real task finishing time of a completed task, and $\tilde{EFT}_j$ is the estimated EFT. The pseudo code of DTS is shown in Algorithm 2.

VII. DATASET EVALUTION

A. Hardware Implementation and Dataset

Cloud server. We evaluate SkyNet on public datasets using only the cloud server. The cloud server is implemented on a server running an Ubuntu 18.04 system with Intel Xeno Silver 4210 @2.20GHz CPUs and NVIDIA RTX 2080Ti GPUs with 12 GB of memory. For detection, we use YOLOX-Tiny [16] as the person detector and RetinaFace-10g [14] as the face detector. We use ArcFace based on ResNet50 [1] as the feature extractor. The FWN is implemented by Pytorch 1.8.0.

Dataset. We evaluate SkyNet on the following two public datasets: i) CFP dataset [22], which provides a total of 7000 multi-view images of 500 identities, each with 10 frontal and 4 side images. ii) DroneFace dataset [23], contains 2057 images of 11 objects, including 5 sets, 620 original images, 1364 frontal images and 73 portrait images. This dataset is chosen because of the raw images captured from different distances and depression angles to simulate the FoV of a drone.

B. Evaluation Results on Public Dataset

1) Evaluation on CFP dataset: **Accuracy:** SkyNet uses 3 input channels for person recognition, corresponding to 3 images of a person. We compare SkyNet with two baselines: i) *RetinaFace+ArcFace (1 image)*: it recognizes each image and

Algorithm 2 Dynamic Task Scheduling Algorithm (DTS)

Input: $Queue_a$, N, each D_i, M, each E_j:[CPU_j, GPU_j, $Queue_j$, $Delay_j$]

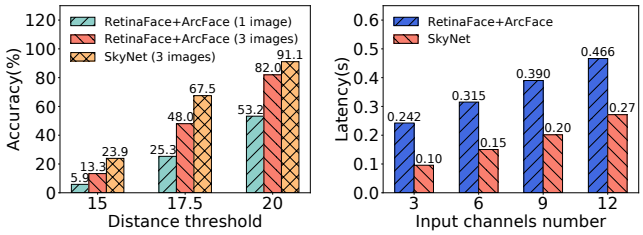
while DTS is running **do**
  assignments $\leftarrow \{\}$
  while $Queue_a$ is not empty **do**
    $taskId \leftarrow Queue_a.front()$
    for j in [1,M] **do**
      $EFT_j^{taskId} \leftarrow MLR(Delay_j, ||Queue_j||, CPU_j, GPU_j)$
    end for
    if $\min EFT_j^{taskId} <$ *shooting interval* **then**
      $DD \leftarrow \arg\min_j EFT_j^{taskId}$
    else
      $DD \leftarrow E_0$
    end if
    push (*taskId*, *DD*) into *assignments*
    push *taskId* into $Queue_{DD}$
  end while
  for i in [1,N] **do**
    sync *assignments* with D_i
  end for
end while

gives a separate recognition result; ii) *RetinaFace+ArcFace (3 images)*, it first calculates the feature distance between each image and the target person image, and then for a person, only the recognition result of the image with the smallest distance is reserved as the recognition result of this person. This means taking the best of the three recognition results for a person. In the experiment, a person has three images, called an *image set*. As shown in Figure 7a, the accuracy of SkyNet is higher than the two baseline methods at all different distance thresholds. When the distance threshold is set to 20, the accuracy of SkyNet is 91.08%, which is 37.9% and 9.1% higher than the two baselines, respectively. This shows that the multi-view face fusion scheme captures more feature information of a person's face, which can improve face recognition accuracy.

Latency: We compare the computational latency of SkyNet and the baseline under the different number of input channels, i.e., the number of images in *image set*. Figure 7b shows that the latency of SkyNet is only about half of the baseline. This is because SkyNet benefits from the PHC and can efficiently utilize the computing resources of devices.

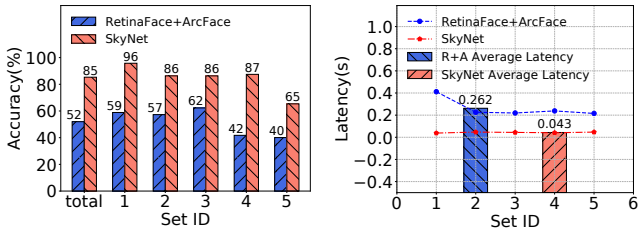
2) Evaluation on DroneFace dataset: **Accuracy:** We compare the performance of SkyNet with the baseline (*RetinaFace+ArcFace*) in recognizing images in the DroneFace dataset. Figure 8a shows the accuracy of SkyNet and the baseline on different image sets. On each set, SkyNet shows better performance. Overall, SkyNet is 33% more accurate than the baseline.

Latency: Similarly, we compare the latency of SkyNet and the baseline. As shown in Figure 8b, the average latency of SkyNet is 83.5% lower than the baseline, consuming only 0.043s.



(a) Accuracy comparison (b) Latency comparison

Fig. 7: Performance Evaluation on CFP dataset.



(a) Accuracy comparison (b) Latency comparison

Fig. 8: Performance Evaluation on DroneFace dataset.

VIII. REAL-WORLD EVALUATION

A. Hardware Implementation and Real-flight Experiment

Drones. We use four F450 quadcopters in the experiment. Each drone is equipped with a PIXHAWK [24] 2.4.8 flight control, an M8N GPS [25], a 4K action camera and an AI edge computing platform, NVIDIA Jetson Xavier NX. For the drone's on-board pipeline, we use YOLOX-Tiny [16] as the person detector and RetinaFace-10g [14] as the face detector.

Edge devices. We use three NVIDIA Jetson Xavier NX as the three edge devices and ArcFace [1] as the feature extractor. FWN is implemented using Pytorch 1.8.0. The implementation of the cloud server is the same as Section VII. Data transfer between devices via WLAN (WiFi protocol 802.11 ac) at 18 Mbps upload/download rate.

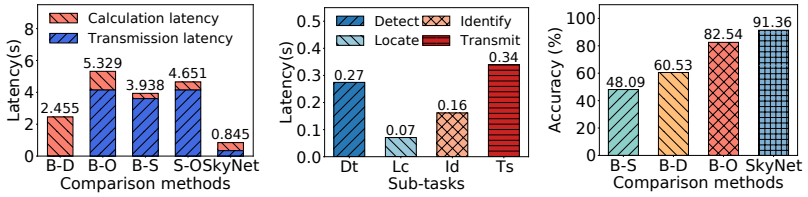
Real-flight Experiment. We deploy and evaluate SkyNet on drones, edge devices, and the cloud server in real-world experiments. In two scenes with 20 people moving freely indoors at 81m^2 and outdoors at 554m^2, we use three drones to capture 4K images and one drone to capture 1080p images, and each experiment is performed for 5 minutes. The four drones are located at the four corners of each scene, 5m above the ground indoors and 10m above the ground outdoors. We run a total of nine experiments, including five indoor experiments and four outdoor experiments.

B. SkyNet Overall Performance

We first evaluate the overall performance of SkyNet. At a once-per-second shooting instant, each drone takes a photo of the crowd. We define the entire operation flow of locating and recognizing the target person to be found as a *task*.

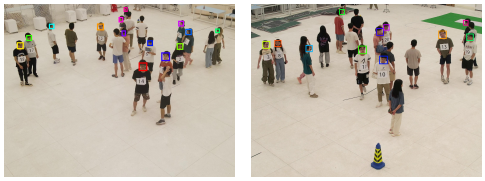
1) Baseline Methods:

- Baseline of single drone (B-D). This baseline runs RetinaFace for face detection and ArcFace for face recognition using a single drone with on-board computing power.



(a) Task latency comparison (b) Latency of sub-tasks (c) Accuracy comparison

Fig. 9: The Overall Performance of SkyNet.



(a) Drone view on 45° (b) Drone view on 225°

Fig. 10: On-Board Computing Output Example.

- Baseline with full offloading (B-O). In this baseline, four drones transmit images to an edge device. The edge device then runs the identification model to find all the people in the drone's view and calculate the feature distance between each person and the target person. If the shortest feature distance is lower than the threshold, the corresponding person is regarded as the target person.
- Baseline with down sampling (B-S). This baseline is different from B-O because the drone transmits the down sampled image (640×640) to the edge device.
- SkyNet with full offloading (S-O). SkyNet runs without DTSH, i.e., the cloud server always serves as the DD.

2) Evaluation Metrics:

- Task Latency: the time spent executing a *task*, including transmission latency and computational latency.
- Accuracy: Top-1 identification accuracy at a specific False Accept Rate (FAR), e.g., FAR=10^{-6}.

3) Evaluation Results: The overall performance evaluation of SkyNet is shown in Figure 9.

Task Latency: The latency of SkyNet is much smaller than all baselines. As shown in Figure 9a, SkyNet is 6.31 times faster than the B-O baseline, which transmits 4K images with a transmission latency of up to 4.154 seconds. SkyNet is also 2.91 times faster than the B-D baseline. Due to the limited computing resources of drones, B-D has a large computational latency, reaching 2.455s.

We further investigate the cost of each sub-task (i.e., detection, localization, recognition, and transfer) in the SkyNet task pipeline. As shown in Figure 9b, among the computational latency, the detection sub-task has the largest latency, while the optimized localization and recognition sub-tasks have less latency. It can be found that the transmission latency is greater than the computational latency. SkyNet with task scheduling effectively utilizes the computing resources of multiple devices, thereby reducing the computational latency. More importantly, because only face sub-images are transmitted, the transmission latency is also greatly reduced.

Accuracy: As shown in Figure 9c, SkyNet has the highest accuracy of 91.36%, which is 8.82%, 30.83% and 43.27% higher than the B-O, B-D, and B-S baselines, respectively. The accuracy of the B-S baseline is the lowest because it loses much information during down sampling. Figure 10 shows two images taken by two drones with the face bounding boxes generated by drone on-board computations. In SkyNet, full-resolution face sub-images are transferred to the DD. The multi-view fused face feature provides more information than any single-view face feature. Even if a person's face cannot be detected in a drone's FoV (e.g., a person with his/her back to the drone), his/her identity can still be identified because other drones may be able to capture his/her face.

C. Evaluation of SkyNet's Localization Performance

We evaluate the localization performance of SkyNet in 5 experiments with different numbers of people. We calculate the localization error, i.e., the error between the MDPL output position and the true position, and the latency in completing the localization task.

Localization error: As shown in Figure 11a, the average error of each task collection is about 18.65cm. Figure 11b visually shows the comparison of the two real tracks of the experimental participant and a series of consecutive positions output by SkyNet in one minute.

Localization latency: As shown in Figure 11a, the latency of MDPL increases with the number of people. The MDPL latency for twenty people is 0.071 seconds.

D. Evaluation of SkyNet's Scheduling Performance

We analyze the impact of the DTSH module to evaluate the performance improvement brought by scheduling.

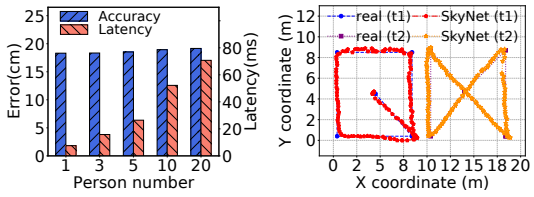
Latency: We compare the latency of SkyNet with and without the DTSH module when processing different numbers of *tasks*. As shown in Figure 12a, the latency of SkyNet with the DTSH module is at least about 15% lower than that of SkyNet without the DTSH module. Note that the average of this latency is larger than the task latency because it includes both the task latency and the queuing latency, and what the DTSH does is reduce queuing latency.

Device queue length: Given 100 tasks, as shown in the figure 12b, if SkyNet runs without the DTSH module, all tasks are assigned to the cloud server (device0 in the figure). If SkyNet runs with the DTSH module, the 100 tasks are assigned almost evenly to each device. Figure 12c shows the length of the task queue for each device over time. When SkyNet runs with the DTSH module, each device has a shorter task queue.

E. Effect of Different Parameters

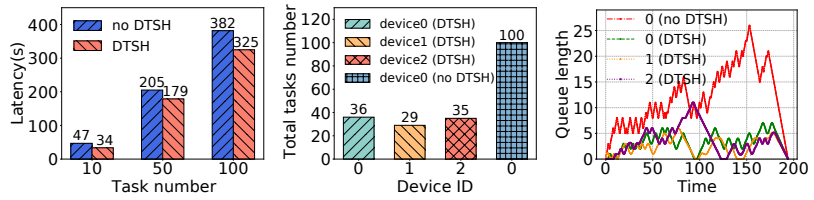
We analyze the effect of the crowd size and the drone number on SkyNet's accuracy and latency. We run SkyNet on different-sized crowds using different numbers of drones.

Latency: As shown in Figure 13a, using more drones leads to higher task latency. As the number of people increases, the end-to-end latency increases due to the increase in sub-images to be processed and data to be transmitted due to



(a) Localization Performance (b) Track visualization of localization results.

Fig. 11: Localization Performance Evaluation.

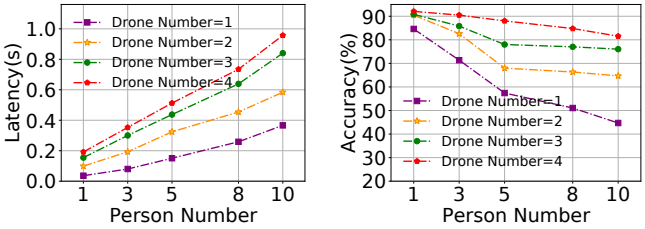


(a) Effect of DTSH on latency (b) Total number of tasks (c) Workload visualization of each device.

Fig. 12: Scheduling Performance Evaluation.

computational offloading. task latency increases faster when the crowd size exceeds 5 people.

Accuracy: As shown in Figure 13b, larger crowds lead to lower accuracy due to mutual occlusion between people. Using more drones can alleviate the problem of accuracy degradation as more details are captured. When the number of drones exceeds 3, the performance gain from adding drones decreases.



(a) Effect on latency (b) Effect on accuracy

Fig. 13: Effect of Different Parameters.

IX. Related Work

A. Face Identification

Computer vision techniques have been applied to drones to detect faces [26]. Erina et al. [27] propose a method for face recognition from a drone view. Hwai-Jung Hsu et al. [28] propose that $Face^{++}$ are limited by the height, distance and depression angle of drones. Triantafyllidou et al. [29] design a lightweight CNN for face detection on drones. Amato et al. [30] focus on the impact of face multi-resolution on face recognition. There are some projects aimed at improving the performance of face recognition in normal scenes and completing high-precision face recognition [31].

One main challenge in face recognition using multiple drones is aligning multi-view information [32]. Many studies of multi-view object alignment use carefully selected feature markers to extract object features and match them into another view, such as SIFT [33], SURF [34], and HOG [35]. These studies use machine learning models such as SVM [36], Adaboost [37] for feature matching, and use the sliding window technique [38] to search images of other views, with high matching complexity and computational latency. These limit their effectiveness and applicability.

B. Localization and tracking

Most 3D localization solutions acquire additional sensory information through specialized devices such as stereo cameras and LiDARs. Knoppe et al. [39] propose a drone system with a stereo camera that collects spectral image patches. Wang et al. [40] utilize the collaboration of drones equipped with multiple-input multiple-output radar to locate marine targets based on triangulation. ORB-SLAM [41] achieves tracking across video frames by extracting feature points from sparse point clouds. The reliance on specialized equipment makes these solutions expensive and difficult to deploy widely.

Object tracking is critical for scenarios that require continuous targeting of objects, such as capture and child searches. Silva et al. propose a face recognition and tracking system [42], in which the same person in different video frames is re-identified based on the face embedding vectors obtained through CNN. However, the target's position is relative to image coordinates rather than world coordinates, making it difficult to accurately track targets in the real world.

X. Conclusion

In this paper, we propose SkyNet, a multi-drone cooperation framework for accurate and real-time identification and localization. SkyNet can accurately locate a person in 3D real world using only conventional 2D cameras and can align the face sub-images of one person from different drone views. To improve identification accuracy, we design a novel fusion identification pipeline, which exploits images from different views by fusing them according to weights reflecting legibility. SkyNet can achieve real-time localization and identification through its ability of dynamic task scheduling. We implement and evaluate SkyNet in real life, and the result shows that SkyNet achieves 91.36% identification accuracy, less than 0.18m localization error, and less than 0.84s latency.

XI. Code

To make it easier for readers to understand and deploy SkyNet, we provide public access to the code at https://doi.org/10.5281/zenodo.7467108.

Acknowledgment

This work is supported by National Natural Science Foundation of China under grant No. 61972189, the Major Key Project of PCL under grant No. PCL2021A03-1, Shenzhen Science and Technology Innovation Commission: Research Center for Computer Network (Shenzhen) Ministry of Education, and the Shenzhen Key Lab of Software Defined Networking under grant No. ZDSYS20140509172959989.

REFERENCES

[1] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019, pp. 4685–4694.

[2] J. Lezama, Q. Qiu, and G. Sapiro, "Not afraid of the dark: Nir-vis face recognition via cross-spectral hallucination and low-rank embedding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Jul. 2017, pp. 6628–6637.

[3] C.-H. Chu and Y.-K. Feng, "Study of eye blinking to improve face recognition for screen unlock on mobile devices," *Journal of Electrical Engineering and Technology*, vol. 13, no. 2, pp. 953–960, Mar. 2018.

[4] S. Girmay, F. Samsom, and A. M. Khattak, "Ai based login system using facial recognition," in *2021 5th Cyber Security in Networking Conference (CSNet)*. IEEE, Oct. 2021, pp. 107–109.

[5] S. Li, X. Ning, L. Yu, L. Zhang, X. Dong, Y. Shi, and W. He, "Multi-angle head pose classification when wearing the mask for face recognition under the covid-19 coronavirus epidemic," in *2020 international conference on high performance big data and intelligent systems (HPBD&IS)*. IEEE, May. 2020, pp. 1–5.

[6] H.-M. Hsu, Y. Wang, and J.-N. Hwang, "Traffic-aware multi-camera tracking of vehicles based on reid and camera link model," in *Proceedings of the 28th ACM International Conference on Multimedia*, Oct. 2020, pp. 964–972.

[7] Z. Tang, M. Naphade, M.-Y. Liu, X. Yang, S. Birchfield, S. Wang, R. Kumar, D. Anastasiu, and J.-N. Hwang, "Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2019, pp. 8797–8806.

[8] H. P. D. Nguyen and D. D. Nguyen, "Drone application in smart cities: The general overview of security vulnerabilities and countermeasures for data communication," *Development and Future of Internet of Drones (IoD): Insights, Trends and Road Ahead*, pp. 185–210, 2021.

[9] L. B. Neto, F. Grijalva, V. R. M. L. Maike, L. C. Martini, D. Florencio, M. C. C. Baranauskas, A. Rocha, and S. Goldenstein, "A kinect-based wearable face recognition system to aid visually impaired users," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 1, pp. 52–64, Feb. 2017.

[10] S. Panchanathan, S. Chakraborty, and T. McDaniel, "Social interaction assistant: a person-centered approach to enrich social interactions for individuals with visual impairments," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 5, pp. 942–951, Aug. 2016.

[11] "Intel depth camera d455," https://store.intelrealsense.com/buy-intel-realsense-depth-camera-d455.html, 2022.

[12] Y. W. Kuan, N. O. Ee, and L. S. Wei, "Comparative study of intel r200, kinect v2, and primesense rgb-d sensors performance outdoors," *IEEE Sensors Journal*, vol. 19, no. 19, pp. 8741–8750, Oct. 2019.

[13] "Nvidia jetson xavier nx," https://developer.nvidia.com/embedded/jetson-xavier-nx, 2020.

[14] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, "Retinaface: Single-shot multi-level face localisation in the wild," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020, pp. 5202–5211.

[15] "Dji mavic," https://www.dji.com, 2022.

[16] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, Aug. 2021.

[17] M. Najibi, P. Samangouei, R. Chellappa, and L. S. Davis, "Ssh: Single stage headless face detector," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 4885–4894.

[18] X. Tang, D. K. Du, Z. He, and J. Liu, "Pyramidbox: A context-assisted single shot face detector," in *Proceedings of the European conference on computer vision (ECCV)*, Sept. 2018, pp. 797–813.

[19] P. Hu and D. Ramanan, "Finding tiny faces," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 1522–1530.

[20] S. Kim, J. Kim, J. Park, and D. Lee, "Vision-based pose estimation of fixed-wing aircraft using you only look once and perspective-n-points," *Journal of Aerospace Information Systems*, vol. 18, no. 9, pp. 659–664, May. 2021.

[21] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proceedings of the seventh ieee international conference on computer vision*, vol. 1. Ieee, Sept. 1999, pp. 666–673.

[22] S. Sengupta, J.-C. Chen, C. Castillo, V. M. Patel, R. Chellappa, and D. W. Jacobs, "Frontal to profile face verification in the wild," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Jul. 2016, pp. 1–9.

[23] H.-J. Hsu and K.-T. Chen, "Droneface: an open dataset for drone research," in *Proceedings of the 8th ACM on multimedia systems conference*, Jun. 2017, pp. 187–192.

[24] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Pixhawk: A system for autonomous flight using onboard computer vision," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, May. 2011, pp. 2992–2997.

[25] U-blox, "China reports expansion in 5g network coverage," 2021. [Online]. Available: https://www.u-blox.com/en

[26] M. Shahbazi, J. Theau, and P. Menard, "Recent applications of unmanned aerial imagery in natural resource management," *Mapping Sciences & Remote Sensing*, vol. 51, no. 4, pp. 339–365, Jun. 2014.

[27] E. Ferro, C. Gennaro, A. Nordio, F. Paonessa, C. Vairo, G. Virone, A. Argentieri, A. Berton, and A. Bragagnini, "5g-enabled security scenarios for unmanned aircraft: Experimentation in urban environment," May. 2020.

[28] H.-J. Hsu and K.-T. Chen, "Face recognition on drones: Issues and limitations," in *Proceedings of the first workshop on micro aerial vehicle networks, systems, and applications for civilian use*, May. 2015, pp. 39–44.

[29] D. Triantafyllidou, P. Nousi, and A. Tefas, "Fast deep convolutional face detection in the wild exploiting hard sample mining," *Big data research*, vol. 11, pp. 65–76, Mar. 2018.

[30] G. Amato, F. Falchi, C. Gennaro, F. V. Massoli, and C. Vairo, "Multi-resolution face recognition with drones," in *2020 3rd International Conference on Sensors, Signal and Image Processing*, Oct. 2020, pp. 13–18.

[31] "Iarpa-janus," https://www.iarpa.gov/research-programs/janus.

[32] D. Wierzbicki, "Multi-camera imaging system for uav photogrammetry," *Sensors*, vol. 18, no. 8, Jul. 2018.

[33] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. Ieee, 1999, pp. 1150–1157.

[34] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[35] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. Ieee, 2005, pp. 886–893.

[36] V. Vapnik and A. Y. Lerner, "Recognition of patterns with help of generalized portraits," *Avtomat. i Telemekh*, vol. 24, no. 6, pp. 774–780, 1963.

[37] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[38] K. R. Sapkota, S. Roelofsen, A. Rozantsev, V. Lepetit, D. Gillet, P. Fua, and A. Martinoli, "Vision-based unmanned aerial vehicle detection and tracking for sense and avoid systems," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Ieee, Oct. 2016, pp. 1556–1561.

[39] E. Honkavaara, H. Saari, J. Kaivosoja, I. Pölönen, T. Hakala, P. Litkey, J. Mäkynen, and L. Pesonen, "Processing and assessment of spectrometric, stereoscopic imagery collected using a lightweight uav spectral camera for precision agriculture," *Remote Sensing*, vol. 5, no. 10, pp. 5006–5039, Oct. 2013.

[40] X. Wang, L. T. Yang, D. Meng, M. Dong, K. Ota, and H. Wang, "Multi-uav cooperative localization for marine targets based on weighted subspace fitting in sagin environment," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5708–5718, Apr. 2022.

[41] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[42] S. H. Silva, P. Rad, N. Beebe, K.-K. R. Choo, and M. Umapathy, "Cooperative unmanned aerial vehicles with privacy preserving deep vision for real-time object identification and tracking," *Journal of parallel and distributed computing*, vol. 131, pp. 147–160, Apr. 2019.