# Spatio-Temporal Fusion Based Low-Loss Video Compression Algorithm for UAVs with Limited Processing Capability

Qianyuan Zhang, Desheng Wan, Hao Chen, Lianghua Cheng, Jiayi Chen, and Chaocan Xiang(✉)

College of Computer Science, Chongqing University, Chongqing 400044, China
xiangchaocan@cqu.edu.cn

**Abstract.** Real-time urban crowd surveillance is essential for riot supervision, epidemic prevention, and urban emergency management. Unmanned aerial vehicles (UAVs) provide a promising way for real-time crowd surveillance due to their convenient deployment and flexible mobility. However, the limited wireless transmission bandwidth and the large capacity of high-definition video pose great challenges to the real-time transmission of UAV-captured videos. Although existing edge computing-based video compression algorithms can partially solve this dilemma, the complexity of these algorithms makes them inapplicable for edge devices with limited processing capacity. To this end, we propose a lightweight spatiotemporal fusion based low-loss video compression algorithm, which consists of two parts: feature clustering-based temporal sampling and dynamic encoding-based spatial sampling. The first module clips the video content from a temporal perspective by identifying inter-frame redundancy. The second module compresses the video content from a spatial perspective by examining regions of interest (RoIs) within each frame and utilizing background filtering to analyze intra-frame encoding. This lightweight algorithm effectively reduces the size of the video file while maintaining high-quality output, which is compatible with edge devices' constrained process power. The experimental results demonstrate that the proposed algorithm maintains minimal loss in crowd detection accuracy while reducing transmission latency by 31.3%.

**Keywords:** Urban crowd surveillance · UAVs · Constrained processing capabilities · Low-loss video compression · Spatio-temporal fusion

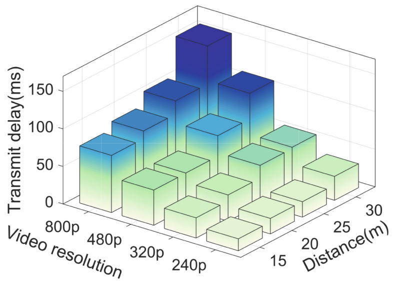
## 1 Introduction

Urbanization has led to a rapid increase in population density, increasing the risk of stampedes and riots in urban areas, which poses a serious threat to public

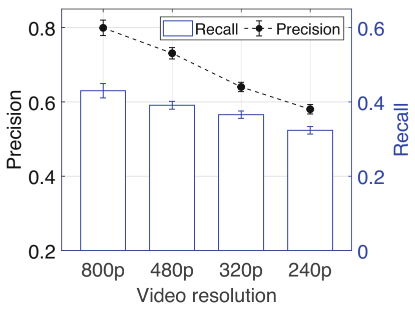
This work is supported by the National Natural Science Foundation of China (62172063).

safety [10]. Therefore, crowd surveillance gains paramount importance in urban safety emergency management. Currently, crowd surveillance mainly relies on fixed cameras deployed in cities [19], but they suffer from blind spots, poor mobility, as well as high deployment and maintenance costs [6]. Other crowd surveillance methods based on WiFi or millimeter-wave radar have also been studied widely. However, they have low accuracy in detecting mobile crowds and limited detection range, making them insufficient to use in practical scenarios.

Fortunately, Unmanned aerial vehicles (UAVs) have become increasingly popular in civil and commercial applications due to their low cost, flexible mobility, and wide range of video capturing capabilities, which provides a promising way for large-scale urban crowd surveillance [24,28]. Although some works have studied UAV-based crowd surveillance systems, they either improve high-precision detection networks that require high-resolution videos, leading to high latency [7–9,33], or only consider real-time performance and use lossy compression on the video, resulting in reduced accuracy [27,32]. Relevant experiments are also conducted to verify the problems:



(a) Transmission Latency



(b) Detection accuracy and recall

**Fig. 1.** Analysis of UAV transmission latency and detection accuracy at different resolutions and transmission distances.

i) High-precision drone surveillance systems are difficult to achieve real-time requirements. As shown in Fig. 1a, When transmitting an 800p high-definition resolution 10-minute video from a drone to a ground station, only 30m away, the latency exceeds 30 min, causing severe lag in crowd surveillance information and making it unable to ensure the real-time safety of urban crowds.
ii) Compressed video leads to a drop in crowd detection accuracy. As illustrated in Fig. 1b, when the transmission distance is fixed, compressing the video frames from 800p to 320p reduces the transmission latency by 65.7%. However, the compression also leads to a sharp drop in detection accuracy from 80% to 64%, making it incapable of meeting the high precision requirements of crowd target detection.

**Table 1.** Comparisons of edge-computing devices [1].

| | | Lightweight devices | | | High-performance devices | |
|---|---|---|---|---|---|---|
| Devices | | RP3 | RP3+ICS | JetsonNano | OrinNX | AGXOrin |
| AI Performance | | - | - | 0.472TFLOPs | 70TOPS | 200TOPS |
| Models (fps) | InveptionV4 | - | - | 11 | 593 | 1337 |
| | TinyYoloV3 | 0.5 | - | 25 | 1156 | 2611 |
| | Unet | - | 5 | - | 183 | 387 |
| | DashcamNet | - | - | 11 | 689 | 1482 |
| Weight (kg) | | 0.1 | 0.2 | 0.241 | 0.76 | 1.5 |
| Power (watt) | | 2 | 2 | 5 | 20 | 40 |

To address the above issues, UAV-mounted lightweight edge devices is utilized to extract effective crowd information from drone-captured video and remove redundant content to achieve low transmission latency and high detection accuracy in drone-based crowd surveillance. However, achieving this goal confronts the following two challenges:

- **Designing lightweight algorithms adapted to edge devices with limited processing capability is challenging.** As depicted in Table. 1, high-performance edge devices have large weight and high power consumption, while the drone (such as S500 [4]) has a mass of only 1.3 kg and a power consumption of 50 W. If equipped with AGXOrin (1.5 kg, 40 W), the UAV's endurance will be reduced by 75% at least. However, the processing power of lightweight devices (e.g., RP3 and JetsonNano) is severely constrained. Therefore, designing lightweight video processing algorithms for limited capabilities is challenging.
- **It is hard to extract effective crowd information and filter out invalid content in drone-captured videos.** Drone footages contain redundant information, including inter-frame repeated content and intra-frame background regions. However, conventional methods for lightweight video compression either fail to utilize image and crowd features related to crowd surveillance tasks, leading to compressed crowd areas inside frame and a decline in detection accuracy [19,29], or only clip between frames or compress inside frames, resulting in a still relatively large processed video volume [22,23]. Hence, efficiently extracting crowd information while filtering out invalid content to minimize latency and achieve high-precision crowd surveillance is a huge challenge.

To address the above challenges, this paper proposes the **Spatiotemporal Fusion based Low-loss Video Compression Algorithm** tailored to edge devices with limited computational capabilities. Specifically, the algorithm consists of two modules: **1) Feature Clustering based Temporal Video Sampling module.** The crowd and scene features are initially extracted from the UAV-captured videos to comprehensively characterize the frames' information relevant to the crowd detection task. Then, using a feature similarity-based clustering algorithm with a sliding window, we clip the redundant video frames and retain keyframes, minimizing the inter-frame similarities. **2) Dynamic Encoding based Spatial Video Sampling module.** For the temporally sampled keyframes, the regions of interest (RoIs) are extracted for video encoding based on lightweight background filtering and target detection in the spatial domain. Subsequently, dynamic video encoding is executed to maximize the reduction of redundant non-RoIs data while maintaining the visual quality of the crowd region. As a result, our algorithm retains only useful data related to crowd surveillance in intra-frames and inter-frames, while irrelevant information is excluded, which satisfies the capability limitations of UAV-end lightweight devices and ensures the minimum video transmission delay and loss rate of target detection.

In summary, this paper makes the following contributions:

1. We present an intelligent UAV-based edge computing system to realize low-latency and high-accuracy crowd surveillance through intelligent collaborative processing between UAVs and ground stations at the edge side.
2. We propose a lightweight spatio-temporal fusion based low-loss video compression algorithm for computing capacity limitation of the UAV end, using crowd and scene features and edge detection information to compress surveillance video from the temporal and spatial aspects to ensure the minimization of video transmission latency and loss rate of target detection.
3. We build the system prototype and give an extensive performance evaluation. The experimental results show that the proposed algorithm can reduce the transmission delay by 31.3% while losing only within 2% of the crowd-counting accuracy.

The rest of the paper is organized as follows. First, the system architecture is presented in Sect. 2. Then we design the spatio-temporal fusion based low-loss video compression algorithm in Sect. 3. In Sect. 4, we conduct extensive experiment evaluations, while implementing the system prototype for real-world application in Sect. 5. Finally, we review related work in Sect. 6 and conclude this paper in Sect. 7.

## 2 System Overview

In this section, we present an overview of the intelligent UAV-based edge computing system for real-time surveillance of urban large-scale crowd gathering. As shown in Fig. 2, it consists of the following two components:

- **UAV module.** This module includes drones and the mounted lightweight edge processing devices, such as Jetson Nano [2]. It performs mobile surveillance and video capturing of crowd areas. The captured videos are then compressed by the lightweight edge device with spatiotemporal low-loss video compression techniques and transmitted to the ground control station in real-time.

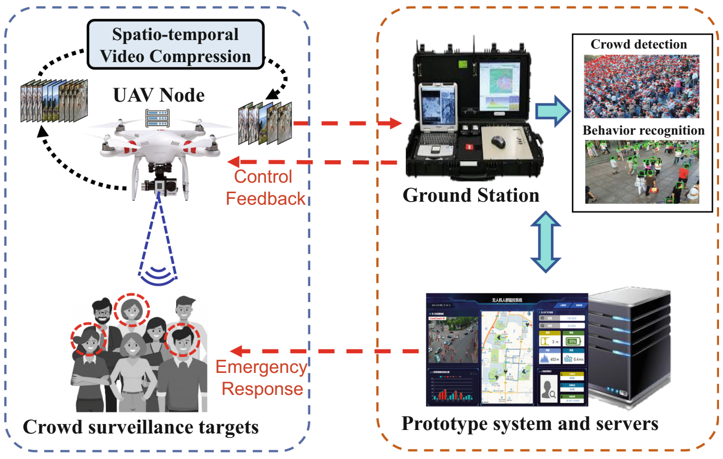


**Fig. 2.** System architecture.

- **Ground Control Station module.** This module comprises a ground control station (GCS) and a user interface for crowd supervision, responsible for controlling the UAV's flight trajectory and processing the compressed videos from the UAV node for the relevant crowd analysis applications. The analyzed results are finally transmitted to the cloud server for data storage and aggregation.

The workflow of the system is as follows: i) Supervisors deploy the ground control station in crowded areas (e.g., commercial plazas and tourist spots, etc.) and set drones' patrol trajectories. ii) The drones capture videos of the gathering crowds while the mounted edge device processes the videos with the spatio-temporal fusion based video compression method proposed in this paper and transmits them to the ground control station with low latency and high precision. iii) The ground control station receives the UAV-captured videos for crowd surveillance applications such as crowd counting [20] and social distancing analysis [24]. Meanwhile, the crowd analysis results are uploaded to the cloud server for data visualization. Besides, the ground control station feedbacks the crowd analysis results (e.g., crowd detection counts, proposed bounding boxes, confidence level, tracking trajectory, etc.) to the UAV node through wireless networks so that the UAV node can refine the detection results for more accurate spatio-temporal video compression.

The key of the intelligent UAV-based edge computing system is the spatio-temporal fusion based low-loss video compression algorithm on the UAV node, satisfying the computational constraints of lightweight edge devices while minimizing the video transmission latency under a specific loss rate of target detection. Hence, the algorithm design will be specified in Sect. 3.

## 3 Algorithm Design

In this section, the lightweight low-loss video compression algorithm based on spatio-temporal fusion sampling is proposed to mitigate the video transmission delay and adapt to the limited processing capability of the edge devices at the UAV node. As illustrated in Fig. 3, it consists of two modules:

1) *Feature clustering based temporal video sampling* (Sect. 3.1): Utilizing the frame clustering based on the crowd feature and scene feature, we clip the redundant frame sequences of UAV-captured video to reduce the number of video frames.

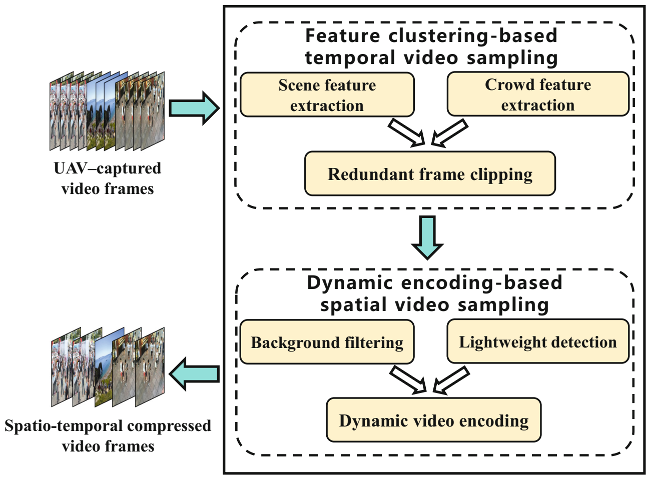

UAV–captured video frames
Feature clustering-based temporal video sampling
Scene feature extraction
Crowd feature extraction
Redundant frame clipping
Dynamic encoding-based spatial video sampling
Background filtering
Lightweight detection
Dynamic video encoding
Spatio-temporal compressed video frames

**Fig. 3.** Low-loss video compression framework based on spatio-temporal sampling fusion.

2) *Dynamic encoding based spatial video sampling* (Sect. 3.2): To further reduce the inefficient information inside a frame, we distinguish regions of interest (RoIs) of the frame and dynamically encode the frame according to the division of the encoding RoIs.

### 3.1 Feature Clustering Based Temporal Video Sampling

This module clips the drone-captured video based on the feature similarity between frames to significantly remove the redundant frames without affecting the accuracy of crowd detection. As depicted in Fig. 4, the crowd and scene features of video frames are first extracted, and then feature clustering and redundant frames are cut according to the feature similarity of the video frames. Finally, the number of redundant frames in the output video frames is significantly reduced compared with the original input video.

**Table 2.** Frequently used notations.

| Notations | Descriptions |
|---|---|
| $I_k$ | The $k_{th}$ frame |
| $N_k$ | The number of bounding box of $k_{th}$ frame |
| $d_i^k$ | The center coordinate of $i_{th}$ bounding box in $k_{th}$ frame |
| $c_i^k$ | The confidence levels of $i_{th}$ bounding box in $k_{th}$ frame |
| $\boldsymbol{m}_k$ | The crowd density map of $k_{th}$ frame |
| $\boldsymbol{P}_k$ | The crowd feature of $k_{th}$ frame |
| $\boldsymbol{F}_k, \boldsymbol{L}_k, \boldsymbol{Y}_k$ | The frequency domain, luminance and contrast feature of $k_{th}$ frame |
| $\boldsymbol{H}_k^R, \boldsymbol{H}_k^G, \boldsymbol{H}_k^B$ | The distribution ratio for R, G, B channels of $k_{th}$ frame |
| $\boldsymbol{S}_k$ | The scene feature of $k_{th}$ frame |
| $\mathcal{V}_{orig}, \mathcal{V}_{out}$ | The input and output frames of temporal video sampling |
| $I_{orig}, I_{out}$ | The input and output frame of spatial video sampling |
| $X_{k_1,k_2}$ | The spectral value of two-dimensional discrete cosine transform |
| $\mathcal{R}_{edge}$, $\mathcal{R}_{det}$ | The output RoIs of the background filtering and lightweight detection |

**Crowd Feature Extraction.** Considering the detection speed and accuracy, we utilize a lightweight network model to extract the crowd features of the image (e.g., the number of people and their location distribution, etc.) as the basis for the similarity determination of the temporal sampling.

Specifically, to adapt to the extremely limited computational power of UAV edge devices (e.g., Jetson Nano), the lightweight network yolov4-tiny [5] is used for crowd detection of the UAV-captured videos. The number of bounding boxes of $k_{th}$ image $I_k$ is denoted as $N_k$, and the center coordinates and the confidence levels of the $i_{th}$ bounding box in the frame are denoted as $d_i^k$ and $c_i^k$, and $c_i^k \in (0, 1]$. Then a Gaussian kernel is applied to convolve with the above crowd detection results to obtain the crowd density map $\boldsymbol{m}_k$ of the $k_{th}$ frame as follows:

$$\boldsymbol{m}_k = \sum_{i=1}^{N} c_i^k \cdot \delta(x - d_i^k) * G_{\delta_i}(x) \tag{1}$$

where $G_{\delta_i}(x)$ is the two-dimensional Gaussian kernel and $\delta(x-d_i^k)$ is the impulse function. Finally, to alleviate the computation burden of the lightweight edge device, the crowd density map $\boldsymbol{m}_k$ is uniformly pooled to obtain the compressed crowd density map $\boldsymbol{P}_k$ with reduced image size. $\boldsymbol{P}_k$ is then used as the crowd feature of that frame. The frequently used notations are shown in Table 2.

**Scene Feature Extraction.** We utilize the image's frequency-domain and structural information as its scene features to determine the variations in the video background during crowd surveillance. When extracting scene features, the frame's low-frequency information is retained to depict the scene contours of UAV-captured videos. In particular, we apply two-dimensional Discrete Cosine Transform (2D-DCT) [13] for each frame according to Eq. (2) and retain the low-frequency region where the spectral energy is concentrated as the low-frequency spectrum.

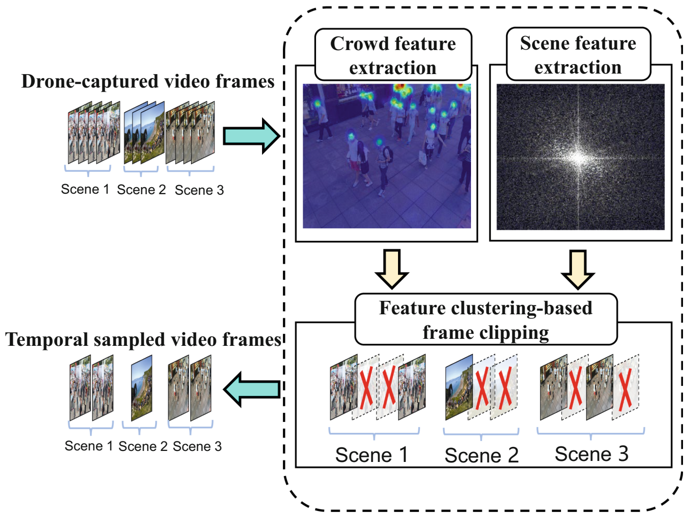


**Fig. 4.** Temporal video sampling.

$$X_{k_1,k_2} = a_{k_1} a_{k_2} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x_{n_1,n_2} \cos\left[k_1 \frac{\pi}{N_1}\left(n_1+\frac{1}{2}\right)\right] \cos\left[k_2 \frac{\pi}{N_2}\left(n_2+\frac{1}{2}\right)\right] \tag{2}$$

where $N_1$ and $N_2$ denote the height and width of the original frame, respectively, and $k_1$ and $k_2$ are the frequency domain coordinates of the 2D-DCT and satisfy $0 \le k_1 \le N_1 - 1$ and $0 \le k_2 \le N_2 - 1$. When $k_1 = 0$, the coefficient $a_{k_1}=1/\sqrt{N_1}$; when $1 \le k_1 \le N_1 - 1$, $a_{k_1}=\sqrt{2/N_1}$. When $k_2 = 0$, the coefficient $a_{k_2}=1/\sqrt{N_2}$; when $1 \le k_2 \le N_2 - 1$, $a_{k_2}=\sqrt{2/N_2}$. According to whether the spectral DCT value is greater than the mean value, each DCT value is quantized to 0 or 1 (i.e., above the DCT mean value is assigned as 1; otherwise, it is assigned as 0). Finally, the quantized result is taken as the frequency domain feature $\boldsymbol{F}_k$ of the $k_{th}$ frame.

Furthermore, the structural information is extracted to capture the structure changes in the scene of the continuous UAV-captured frames. Specifically, the $k_{th}$ frame is uniformly divided into rectangular blocks, and the variance and mean of grayscale values of each block are calculated as the local luminance and contrast. Then the local luminance and contrast values of all blocks are concatenated to form the luminance feature $\boldsymbol{L}_k$ and contrast feature $\boldsymbol{Y}_k$. Next, the pixel distribution ratios of different grayscale levels for the R, G, B channels of the $k_{th}$ frame is denoted as $\boldsymbol{H}_k^R, \boldsymbol{H}_k^G, \boldsymbol{H}_k^B$, which serve as the color distribution features. Finally, the frequency-domain, luminance, contrast, and color distribution features of the $k_{th}$ frame are combined as the scene feature $\boldsymbol{S}_k$.

**Feature Clustering Based Frame Clipping.** The image's crowd and scene features are jointly employed as the combined features, and feature similarity-based clustering is used to extract keyframes and remove redundant ones. The temporal video sampling process is demonstrated in Algorithm 1, mainly involving the following two steps:

**Step 1:** Calculation of inter-frame similarity based on sliding window. Since video frames captured by UAV have similarity within a continuous time range , we use a sliding window to cover consecutive video frames and calculate the cosine similarity as the inter-frame similarity based on the combined features for all frames within the window coverage, according to Eq. (3).

$$\text{Similarity}(X, Y) = \frac{X \cdot Y}{\|X\| \times \|Y\|} \tag{3}$$

**Step 2:** Video frame clustering and redundant frame removal. Based on the inter-frame similarity within the sliding window, the DBSCAN algorithm [16] is applied to feature-based clustering of video frames. Then, based on the clipping threshold, we determine whether to remove each cluster of video frames in the clustering result. Finally, the sliding window is moved in steps, and the feature clustering and redundant frame removal process continues until the sliding window traverses all input frames.

**Algorithm 1:** Feature Clustering based Temporal Video Sampling.

```
Input: UAV-captured continuous video sequence with $N$ frames
       $\mathcal{V}_{orig} = \{I_1, \cdots, I_N\}$
Output: Sampled video sequence with $K$ frames
       $\mathcal{V}_{out} = \{I_i, \cdots, I_j\}, 1 \le i, j \le N$.
1  Initialize: Clipping threshold $\varepsilon = \varepsilon_0$; Sliding window length $L_S = L_{S_0}$; Sliding
   step $d = \lfloor L_S/2 \rfloor$; $\mathcal{V}_{out} = \varnothing$.
2  for $i \leftarrow 0$ to $\lfloor (N-d)/d \rfloor -1$ do
3      for $j \leftarrow 1$ to $L_S$ do
4          for $k \leftarrow 1$ to $L_S$ do
5              Calculate the inter-frame similarity $sim_{i+j,i+k}$ according to Eq. (3)
6          end
7      end
8      $Cls = \text{DBSCANcluster}(\{sim_{i+1,i+1}, \cdots, sim_{i+L_S,i+L_S}\}, \mathcal{V}_{orig})$
9      $\mathcal{V}_{keep} = \text{FrameCut}(Cls, \varepsilon)$
10     $\mathcal{V}_{out} = \mathcal{V}_{out} \cup \mathcal{V}_{keep}$
11 end
12 return $\mathcal{V}_{out}$.
```

## 3.2 Dynamic Encoding Based Spatial Video Sampling

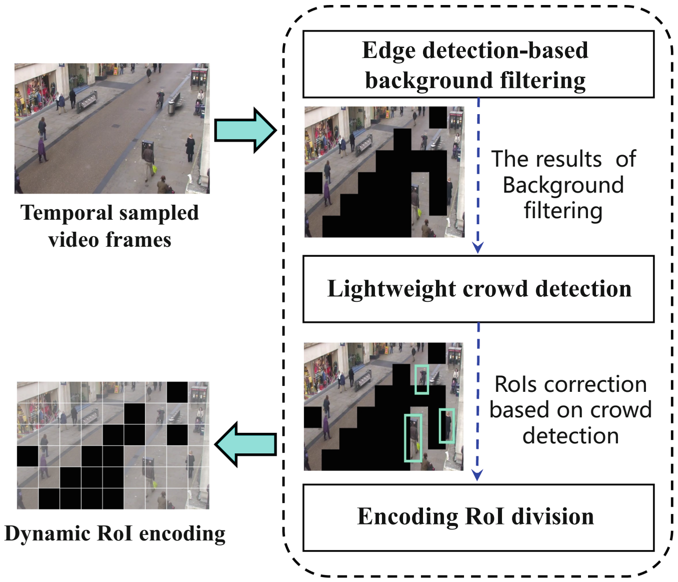

Temporal sampled video frames
Edge detection-based background filtering
The results of Background filtering
Lightweight crowd detection
RoIs correction based on crowd detection
Encoding RoI division
Dynamic RoI encoding

**Fig. 5.** Spatial video sampling.

Although the duplicate content between frames is filtered out via temporal video sampling, each frame still contains many inefficient background regions (such as streets, sky, and other non-crowd areas in crowd surveillance), occupying a large file volume. Therefore, we design the spatial video sampling module based on dynamic encoding to filter out the non-regions of interest (non-RoIs) while maintaining the visual quality of the crowd areas. As shown in Fig. 5, the background regions are filtered out with edge detection and then lightweight crowd detection results are utilized to correct the misclassification of crowd regions as non-RoIs in the edge detection, thus obtaining an accurate division of encoding RoIs. Finally, the frames are dynamically encoded according to the RoI division, i.e., the task-related RoIs are losslessly encoded, while the non-RoIs are lossy encoded.

**Background Filtering Based on Edge Detection.** Compared to the texture and contours in the crowd regions, the edge texture information in the background regions of images is relatively simple. Hence, we utilize the edge texture information of different objects to distinguish and filter the background regions in the image. Lightweight edge detection utilizes the Prewitt operator [13] (Eqs. (4) and (5)). Specifically, for the input image $I_k$, the horizontal and vertical gradients at pixel $(x, y)$ is calculated according to Eqs. (6) and (7):

$$d_x = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{4}$$

$$d_y = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{5}$$

$$G_x = d_x * I_k \tag{6}$$

$$G_y = d_y * I_k \tag{7}$$

where $*$ denotes the 2D convolution operator. Then each element's value $g_{xy}(i,j)$ of the gradient magnitude matrix $G_{xy}$ is calculated as follows:

$$g_{xy}(i,j) = \sqrt{g_x(i,j)^2 + g_y(i,j)^2} \tag{8}$$

where $g_x(i,j)$ and $g_y(i,j)$ represent the values of the horizontal and vertical gradient matrices at position $(i,j)$. If $g_{xy}(i,j) < \lambda_{edge} \cdot \overline{g_{xy}}(i,j)$, the pixel at position $(x,y)$ is labeled as RoI, otherwise it is labeled as non-RoI. The final output RoIs is defined as $\mathcal{R}_{edge} = \{(x,y)|g_{xy}(i,j) \geq \lambda_{edge} \cdot \overline{g_{xy}}(i,j)\}$.

Moreover, to adapt to the limited computation capabilities of the drone nodes, this lightweight edge detector has a time complexity of only $O(HW \cdot \log(HW))$ for edge detection with the image size of $H \times W$, equivalent to a single convolutional layer in a neural network. The algorithm's execution time on actual lightweight edge devices (i.e., Jetson Nano) is less than 3 ms for 800p frame. Hence, this algorithm incurs minimal computational overhead and negligible processing latency at the edge node.

**Lightweight RoI Correction.** Since background filtering based on edge detection relies solely on the intensity of edge textures to distinguish background regions without considering the crowd distribution in the image, it is a common issue of misclassifying crowd areas as background regions [21]. To reduce the computational workload on lightweight edge devices, the crowd detection results in the temporal sampling process is reused to reclassify crowd areas erroneously labeled as background into non-background regions. For the crowd density map $\boldsymbol{m}_k$ obtained in temporal sampling, if an element value in $\boldsymbol{m}_k$ is greater than the maximum value of the detection threshold $e_0$ and the mean value of $\boldsymbol{m}_k$, it is designated as RoI; otherwise, it is labeled as non-RoI. The final output RoIs of the lightweight detection is $\mathcal{R}_{det} = \{(x,y)|m_k(x,y) \geq \max\{e_0, \overline{m_k}\}\}$.

The RoIs from lightweight crowd detection are used to complement and revise the background filtering results, thus the final RoI division is defined as $\mathcal{R}_f = \mathcal{R}_{edge} \cup \mathcal{R}_{det}$.

**Dynamic Quality Encoding.** Upon obtaining the final partition of the video frames into RoIs, we utilize a video encoder for dynamic quality encoding of the frames. The encoding quality map for each frame is dynamically computed based on the RoI partition of the frame, defining the encoding quality for each block within the frame (i.e., RoIs are set to lossless encoding, while non-RoIs are set to lossy encoding). Then, based on the encoding quality map of each frame, the video encoder performs corresponding lossless or lossy encoding for each image block.

**Algorithm 2:** Dynamic Encoding based Spatial Video Sampling.

```
   Input: Coloured video frame $I_{\text{orig}}$ with pixel size $m \times m$
   Output: Spatially sampled video frame $I_{\text{out}}$; Encoding RoI set $\mathcal{R}$.
1  Initialize: Edge detection threshold $\varepsilon_d = \varepsilon_{d_0}$; $\mathcal{R} = \varnothing$; $I_{\text{out}} = I_{\text{orig}}$.
2  $\mathcal{B}$=Division($I_{\text{orig}}$)
3  for $box$ in $\mathcal{B}$ do
4      $\mathcal{R}_{edge}$=EdgeDetect($box$,$\varepsilon_d$)
5      $\mathcal{R}_{det}$=CrowdDetect($box$)
6      if $|\mathcal{R}_{edge}| > 0$ or $|\mathcal{R}_{det}| > 0$ then
7          $\mathcal{R} = \mathcal{R} \cup \{\mathcal{R}_{edge} \cup \mathcal{R}_{det}\}$
8          $I_{\text{out}}$=HighQualityEncode($I_{\text{out}}$, $box$) /* lossless encoding*/
9      else
10         $I_{\text{out}}$=LossyQualityEncode($I_{\text{out}}$, $box$) /* lossy encoding*/
11     end
12 end
13 return $I_{\text{out}}$ and $\mathcal{R}$.
```

The spatial video sampling algorithm based on dynamic encoding is illustrated in Algorithm 2. Its input is $m \times m$ colored video frame $I_{orig}$. Line 1 initializes the parameters. Then we obtain the results of RoI division (line 2). In line 3–12, the edge detection, lightweight crowd detection and dynamic video encoding are executed. At last, the algorithm outputs the dynamically encoded video frame $I_{out}$ with the RoI set $\mathcal{R}$.

To sum up, by jointly incorporating the results of edge filtering and lightweight crowd detection, dynamic encoding-based spatial video sampling achieves lossless encoding for crowd areas while applying lossy compression to the background regions of non-surveillance targets. This module can further reduce video transmission volume without compromising crowd surveillance accuracy.

## 4 Evaluation

### 4.1 Experimental Setup

**Datasets Description.** The drone crowd surveillance datasets used in this paper include The Oxford Town Centre (OTC) [14], Group-detection (GD) [26], the self-labeled drone-captured Multi-scenario Crowd Dataset (MCD), Drone

Vision Challenge dataset VisDrone2019 [12]. All the datasets are compatible with the crowd detection tasks and contain over 20,000 UAV-captured crowd-gathering video frames with resolutions up to 1280 × 720 and above, including multiple scenarios such as crowd riots, large-scale gatherings and city street parades.

**Experimental Methodology and Settings.** Jetson Nano is deployed as lightweight edge processing device on the UAV side and a GPU-equipped laptop is used as a ground base station for data communication via wireless wifi. Deep network inference and video processing is based on TenorRT [3] and FFmpeg. Yolov5 is trained as crowd detection network for GCS with VisDrone2019 dataset on the training server equipped with NVIDIA 3090 GPU.

To show the effectiveness of our algorithm, we evaluate the overall latency, detection accuracy and recall at different crowd scene density and image resolution. The communication distance between the UAV end and the GCS end is set to 30m. Besides, to simulate the scene switching of the real drone-captured video, the video frames of different scenes are mixed and sampled to form the testing set.

**Baseline Methods.** To comprehensively evaluate the performance of edge computing systems, four typical comparison methods with different performances are used in this paper:

- **Un-CVP**: Uncompressed Video Processing method directly transmits the whole UAV-captured video frames to GCS without any compressing. Thus, this method achieves the highest detection accuracy but incurs the largest overall latency.
- **CVP** [19]: Compressed Video Processing method processes all the UAV-captured video frames with lossy compression.
- **CFRC** [30]: Clipped Fixed-RoI Compressing method first clips the UAV-captured video with structural similarity and then compresses frames with fixed RoIs.
- **REMIX** [18]: This algorithm compresses video frames based on adaptive image region segmentation in different compression qualities according to the crowd density of each region.

Furthermore, to evaluate the performance of each module of our algorithm, the following two comparison algorithms are used:

- **TSO**: Temporal Sampling Only algorithm compresses the UAV-captured videos via the temporal sampling module in this paper.
- **SSO**: Spatial Sampling Only algorithm compresses the UAV-captured videos by the spatial sampling module in this paper.

**Evaluation Metrics.** For the experimental studies, we adopt three metrics to evaluate the algorithm performance: 1) **Average Processing Time(APT)**, which is the sum of the average processing delay of the UAV-captured video at the edge of the UAV and the average transmission delay of the compressed video to the ground station per frame; 2) **Precision(P)**, which refers to the proportion of the detection results in the relevant categories to the total returned results, as in Eq. (9); 3) **Recall(R)**, the proportion of relevant categories to the total relevant categories in the detection results, as in Eq. (10).

$$P = \frac{|\mathcal{T}_P|}{|\mathcal{T}_E|} \tag{9}$$

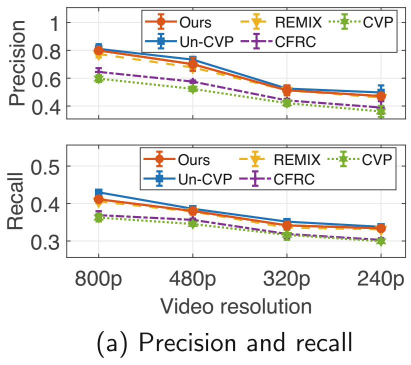
$$R = \frac{|\mathcal{T}_P|}{|\mathcal{T}_G|} \tag{10}$$

where $\mathcal{T}_P$, $\mathcal{T}_E$ and $\mathcal{T}_G$ denote the sets of the correct detection results, the whole detection results, and the ground-truth, respectively.

## 4.2 Experimental Results

**Algorithm Performance Evaluation.** First, we assess the impact of different video resolutions on crowd detection performance and system processing latency. As depicted in Fig. 6a, the video frame resolution gradually decreases from 800p to 240p, and the experiment results show that our algorithm achieves 98.6% detection accuracy and 95.7% recall of the Un-CVP method. Compared to our algorithm, the detection accuracy of the REMIX, CFRC, and CVP methods is, on average, reduced by 2.2%, 17.2%, and 23.2%, respectively. This significant difference in accuracy demonstrates that only our algorithm and REMIX consider crowd information during video processing, ensuring crowd detection accuracy and recall. Moreover, as shown in Fig. 6b, our algorithm reduces the average APT of the Un-CVP and the REMIX methods by 31.3% and 33.25%, respectively.

Next, we evaluate the impact of crowd-gathering scenes in different densities on crowd-detection performance and system processing latency. As shown in



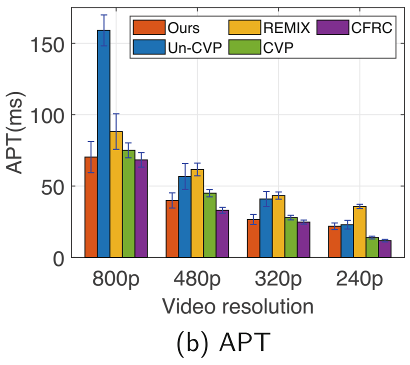
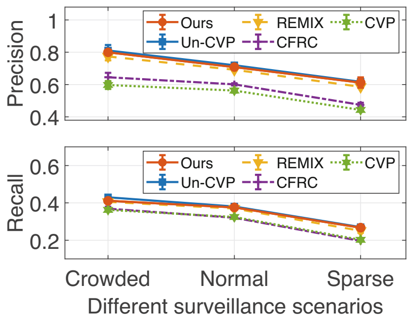
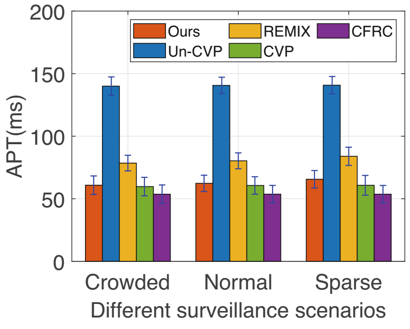


**Fig. 6.** Evaluation of crowd detection performance with different video resolutions, in terms of precision, recall, and latency.



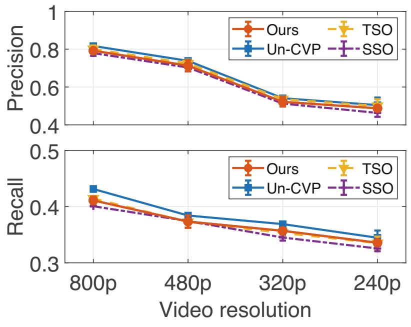
(a) Precision and recall



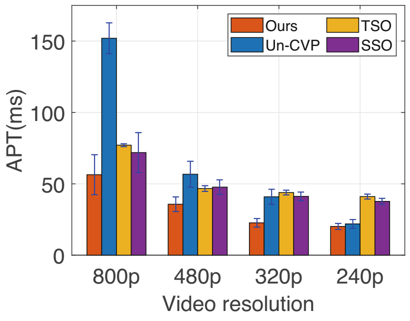
(b) APT

**Fig. 7.** Evaluation of crowd detection performance in different crowd density scenarios, in terms of precision, recall, and latency.

Fig. 7a, when the resolution of different crowd density scenes is set to 800p, our algorithm improves the accuracy by 23.6% and 32.6%, and the recall by 21.6% and 20.5% compared to the CVP and CFRC, while achieving crowd detection accuracy and recall close to the Un-CVP. Furthermore, as shown in Fig. 7b, compared with low-density scenes, our algorithm and the REMIX increase the APT by an average of 5.1% and 4.6%, respectively, in medium and high-density scenes due to the increased delay in processing crowd features. In all crowd density scenes, our algorithm reduces average processing latency by 55.2% and 22.3% compared to the Un-CVP and REMIX methods, respectively, with an average reduction of 77.6 ms and 18.0 ms.



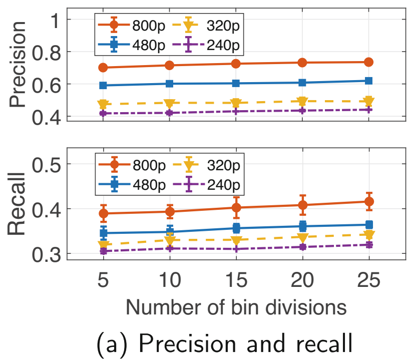
(a) Precision and recall



(b) APT

**Fig. 8.** Evaluation of crowd detection performance with different algorithm modules, in terms of precision, recall, and latency.

**Algorithm's Module Evaluations via Ablation Studies.** We conduct ablation experiments to evaluate the effectiveness of each module in our algorithm.



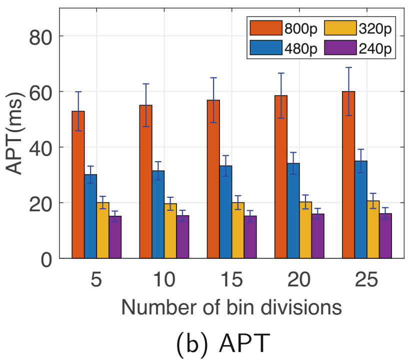


**Fig. 9.** Evaluation of crowd detection performance with different numbers of image divisions, in terms of precision, recall, and latency.

As shown in Fig. 8a, the accuracy of our algorithm, the TSO, and SSO reached 96.7%, 98.6%, and 94.3% of the Un-CVP method, and the recalls reached 96.7%, 96.6%, and 94.6% of the Un-CVP method, respectively. Then, the experimental results of system processing delay, as shown in Fig. 8b, illustrate that our algorithm reduces the average APT of the Un-CVP method by 38.2%. The delay of the TSO and SSO algorithms at higher resolutions (800p and 480p) is reduced by an average of 34.3% and 33.5%, compared to the Un-CVP, while at lower resolutions (320p and 240p), the delay is increased by 36.3% and 47.2%, respectively. This difference in delay can be attributed to the fact that both the TSO and SSO use neural networks, resulting in inherent processing delay that cannot be effectively reduced at lower resolutions. On the other hand, our algorithm's temporal and spatial sampling modules enable the neural network's time consumption to be evenly distributed, reducing the APT remarkably.

**Algorithm's Parameter Evaluations.** We evaluate the impact of the block division parameter *bin* on the algorithm's performance. As depicted in Fig. 9a and Fig. 9b, when the number of image block division *bin* increases from 5 to 25, the experimental results show that with the increase of the bin, the detection accuracy and recall are improved by 2.6% and 2.1%, and the APT is increased by 3.4 ms on average. As a result, increasing the *bin* can improve the detection performance to some extent but also lead to a growth of the APT.

Additionally, our algorithm is tested on video frames of four large-scale gatherings scenarios. As illustrated in Fig. 10, the scenes in column 1 to 4 are crowded city streets, election rallies, square clusters, and mass crowd gatherings of marathons, respectively. The detection results demonstrate that the crowd detection performance of compressed images with the CVP method significantly decreases, while the detection performance of video frames processed by our algorithm is similar to the uncompressed video frames (Un-CVP) in all scenarios.

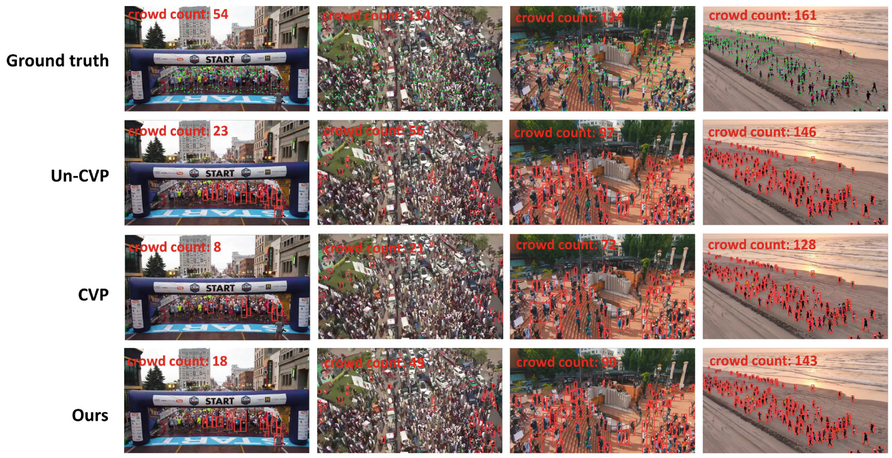


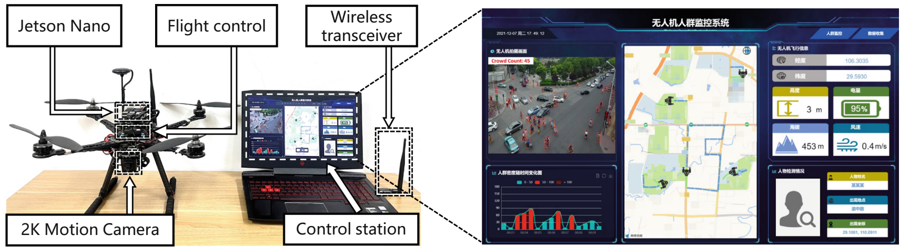
**Fig. 10.** Illustration of crowd detection results in large-scale gatherings scenarios.

In summary, the above experimental results further validate the effectiveness of our algorithm for real-time high-precision video compression in crowd surveillance. In comparison with baseline methods, our algorithm achieves remarkable improvement in reducing video transmission latency by 31.3% while maintaining the detection accuracy loss within 2%. Also, the performance of our algorithm is stable and robust, making it adaptable to crowd supervision in various surveillance scenarios.

## 5 Implementation

To validate the effectiveness and feasibility of our design, we build a prototype UAV-based urban crowd surveillance system. As shown in Fig. 11(a), we utilize an S500 drone [4] equipped with a Jetson Nano lightweight processing device as an edge node. The drone also carries a 2k motion camera, flight control, and a wireless transceiver to capture and stream HD video. And a GPU-equipped laptop is used as the ground station, communicating with the drone in real time via wireless wifi.

The prototype system practically executes crowd surveillance tasks in the following steps: 1) we deploy the ground station near crowded places in urban areas and set up patrol areas for UAVs; 2) the UAVs fly around to capture videos of gathering crowds while the edge devices on board UAVs perform the spatio-temporal low-loss video compression and transmit the compressed videos to the ground side with low latency; 3) the ground station conducts crowd surveillance applications, such as crowd counting and density analysis on the UAV-captured videos, and uploads the results to a cloud server for visualization, as depicted in Fig. 11(b), the user interface of the system displays real-time streaming video, detection results of crowds, the UAVs' flight trajectories and status, specific person recognition results, and crowd density information.



(a) UAV-based crowd-surveillance prototype system (b) System Interface

**Fig. 11.** Intelligent UAV-based real-time crowd surveillance prototype system.

## 6 Related Work

Some work has been done on UAV-based crowd surveillance systems and lightweight video compression algorithms.

**UAV-Based Crowd Surveillance Systems.** UAVs are often combined with crowd-surveillance methods for urban security applications [17,24,25] due to their small size, low cost, and high safety. Singh *et al.* [25] used ScatterNet for human pose estimation to identify violent behavior from the UAV-captured footage. Castellano *et al.* [9] embedded cameras and GPUs in UAVs for real-time crowd density estimation. Woźniak *et al.* [31] used specially designed UAVs with large edge devices to run deep neural networks for crowd surveillance, reducing endurance and difficulty applying to conventional UAVs. As UAV technology continues to advance, its applications in various fields have further expanded. However, due to the limitations of endurance and size, UAVs cannot carry large-scale processing devices, making it difficult to handle complex computing tasks. Therefore, solving the problem of limited UAV edge computing capabilities has become the focus of current research.

**Lightweight Video Compression Algorithms.** Due to edge devices' extremely limited computing resources, lightweight video compression algorithms have been widely studied to solve such problems. Lu *et al.* [22] utilized optical flow motion information and autoencoder networks for efficient video encoding. Cohen *et al.* [11] proposed a modified entropy-constrained quantizer design algorithm for lightweight computing between clouds and edges. He *et al.* [15] designed the overfitted repair neural network (ORNN) to obtain overfitted images. Park *et al.* [23] proposed a fast decision scheme for lightweight neural networks using multi-type trees (MTT) to reduce encoding complexity. Although lightweight video compression algorithms partly solves the video transmission latency, some still have high-performance requirements making it difficult to run on lightweight devices. In addition, the lightweight video compression methods currently are not adapted to crowd detection tasks. Accordingly, achieving high-precision and low-latency drone-captured video compression is still an open problem.

Unlike previous studies, this paper proposes a spatio-temporal low-loss video compression algorithm running on a capability-limited UAV node. It performs inter-frame comprehensive feature extraction and similarity clustering based on video clipping and RoI-based dynamic encoding of videos within frames, ensuring the accuracy of crowd detection and reducing video transmission latency.

## 7 Conclusion

This paper proposes an intelligent UAV-based edge computing system for real-time urban crowd surveillance. The system aims to address the high latency in video transmission and the low accuracy of compressed videos in UAV-based crowd surveillance with limited processing capabilities. Specifically, we propose a spatio-temporal fusion algorithm for low-loss video compression tailored for UAVs with limited edge capabilities. By integrating feature-based clustering for temporal sampling and dynamic encoding for spatial sampling, the algorithm achieves low-loss compression of UAV-captured videos while minimizing transmission latency under a certain detection information loss rate. The compressed videos are transmitted with low latency from the UAV to the ground station, enabling real-time and accurate UAV-based crowd surveillance. Furthermore, based on the actual UAV, ground station, and cloud server, we build a prototype UAV-based edge intelligent system and conduct comprehensive and in-depth experimental evaluations. The experimental results demonstrate that the proposed approach effectively reduces transmission latency while meeting high accuracy requirements.

## References

1. Jetson benchmarks (2023). https://developer.nvidia.com/embedded/jetson-benchmarks
2. Nvidia jetson nano (2022). https://www.nvidia.cn/autonomous-machines/embedded-systems/jetson-nano/
3. Nvidia tensorrt (2022). https://developer.nvidia.com/tensorrt
4. S500 v2 kit (2022). https://www.holybro.com/product/pixhawk4-s500-v2-kit/
5. Yolov4-tiny (2022). https://github.com/AlexeyAB/darknet
6. Cctv: Too many cameras useless, warns surveillance watchdog tony porter (2019). https://www.bbc.com/news/uk-30978995
7. Bai, H., Wen, S., Gary Chan, S.H.: Crowd counting on images with scale variation and isolated clusters. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pp. 18–27 (2019)
8. Castellano, G., Castiello, C., Cianciotta, M., Mencar, C., Vessio, G.: Multi-view convolutional network for crowd counting in drone-captured images. In: Bartoli, A., Fusiello, A. (eds.) ECCV 2020. LNCS, vol. 12538, pp. 588–603. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-66823-5_35
9. Castellano, G., Castiello, C., Mencar, C., Vessio, G.: Crowd counting from unmanned aerial vehicles with fully-convolutional neural networks. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2020)

10. CNN: South Korean authorities say they had no guidelines for halloween crowds, as families grieve 156 victims (2022). https://edition.cnn.com/2022/10/31/asia/seoul-itaewon-halloween-mourning-memorial-intl-hnk/index.html
11. Cohen, R.A., Choi, H., Bajić, I.V.: Lightweight compression of neural network feature tensors for collaborative intelligence. In: 2020 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6. IEEE (2020)
12. Du, D., et al.: VisDrone-DET2019: the vision meets drone object detection in image challenge results. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pp. 213–226 (2019)
13. Gonzalez, R.C.: Digital Image Processing. Pearson Education (2009)
14. Harvey, A., LaPlace, J.: Megapixels: origins, ethics, and privacy implications of publicly available face recognition image datasets. Megapixels **1**(2), 6 (2019)
15. He, G., et al.: A video compression framework using an overfitted restoration neural network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 148–149 (2020)
16. Ionescu, R.T., Smeureanu, S., Popescu, M., Alexe, B.: Detecting abnormal events in video using narrowed normality clusters. In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1951–1960. IEEE (2019)
17. Jha, S., et al.: Visage: enabling timely analytics for drone imagery. In: Proceedings of the 27th Annual International Conference on Mobile Computing and Networking, pp. 789–803 (2021)
18. Jiang, S., Lin, Z., Li, Y., Shu, Y., Liu, Y.: Flexible high-resolution object detection on edge devices with tunable latency. In: Proceedings of the 27th Annual International Conference on Mobile Computing and Networking, pp. 559–572 (2021)
19. Jiang, Y., Miao, Y., Alzahrani, B., Barnawi, A., Alotaibi, R., Hu, L.: Ultra large-scale crowd monitoring system architecture and design issues. IEEE Internet Things J. **8**(13), 10356–10366 (2021)
20. Liu, J., Gao, C., Meng, D., Hauptmann, A.G.: Decidenet: counting varying density crowds through attention guided detection and density estimation. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5197–5206. IEEE (2018)
21. Liu, L., Li, H., Gruteser, M.: Edge assisted real-time object detection for mobile augmented reality. In: The 25th Annual International Conference on Mobile Computing and Networking, pp. 1–16 (2019)
22. Lu, G., Zhang, X., Ouyang, W., Chen, L., Gao, Z., Xu, D.: An end-to-end learning framework for video compression. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3292–3308 (2020)
23. Park, S.H., Kang, J.W.: Fast multi-type tree partitioning for versatile video coding using a lightweight neural network. IEEE Trans. Multimedia **23**, 4388–4399 (2020)
24. Rezaee, K., Mousavirad, S.J., Khosravi, M.R., Moghimi, M.K., Heidari, M.: An autonomous UAV-assisted distance-aware crowd sensing platform using deep shuffleNet transfer learning. IEEE Trans. Intell. Transp. Syst. **23**(7), 9404–9413 (2022)
25. Singh, A., Patil, D., Omkar, S.: Eye in the sky: Real-time drone surveillance system (dss) for violent individuals identification using scatternet hybrid deep learning network. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1629–1637 (2018)
26. Solera, F., Calderara, S., Cucchiara, R.: Socially constrained structural learning for groups detection in crowd. IEEE Trans. Pattern Anal. Mach. Intell. **38**(5), 995–1008 (2015)

27. Sun, J., Li, B., Jiang, Y., Wen, C.Y.: A camera-based target detection and positioning UAV system for search and rescue (SAR) purposes. Sensors **16**(11), 1778 (2016)
28. Wang, Y., Su, Z., Xu, Q., Li, R., Luan, T.H.: Lifesaving with RescueChain: energy-efficient and partition-tolerant blockchain based secure information sharing for UAV-aided disaster rescue. In: IEEE INFOCOM 2021-IEEE Conference on Computer Communications, pp. 1–10. IEEE (2021)
29. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
30. Wang, Z., Simoncelli, E.P., Bovik, A.C.: Multiscale structural similarity for image quality assessment. In: The Thrity-Seventh Asilomar Conference on Signals, Systems and Computers, 2003, vol. 2, pp. 1398–1402. IEEE (2003)
31. Woźniak, M., Siłka, J., Wieczorek, M.: Deep learning based crowd counting model for drone assisted systems. In: Proceedings of the 4th ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond, pp. 31–36 (2021)
32. Xiao, L., Ding, Y., Huang, J., Liu, S., Tang, Y., Dai, H.: UAV anti-jamming video transmissions with QoE guarantee: a reinforcement learning-based approach. IEEE Trans. Commun. **69**(9), 5933–5947 (2021)
33. Yang, Y., Li, G., Wu, Z., Su, L., Huang, Q., Sebe, N.: Reverse perspective network for perspective-aware object counting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4374–4383 (2020)