Crowd Flow Detection from Drones with Fully Convolutional Networks and Clustering

Giovanna Castellano, Corrado Mencar, Gaetano Sette, Francesco Saverio Troccoli and Gennaro Vessio
Department of Computer Science, University of Bari Aldo Moro, Bari, Italy
{giovanna.castellano, corrado.mencar, gennaro.vessio}@uniba.it

2022 International Joint Conference on Neural Networks (IJCNN) | 978-1-7281-8671-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/IJCNN55064.2022.9891954

***Abstract*—Crowd analysis from drones has attracted increasing attention in recent times, thanks to the ease of deployment and affordable cost of these devices. However, how this technology can provide a solution to crowd flow detection is still an explored research question. In this paper, we contribute by proposing a crowd flow detection method for video sequences shot by a drone. The method is mainly based on a Fully Convolutional Network model for crowd density estimation, which aims to provide a good compromise between effectiveness and efficiency, and clustering algorithms aimed at detecting the centroids of high-density areas in density maps. The method was tested on the VisDrone Crowd Counting dataset—characterized not by still images but by video sequences—providing promising results. This direction may open up new ways of analyzing high-level crowd behavior from drones.**[1]

***Index Terms*—crowd counting, crowd density estimation, crowd flow detection, drones, computer vision, deep learning**

I. Introduction

Due to population growth and the increasing degree of urbanization, more and more people live in urban areas. The advantages of this trend are the enrichment of cultural life and the full use of the convenient urban infrastructure. At the same time, gatherings of people, which can arise for various reasons, such as political rallies, festival celebrations, concerts, and so on, pose serious challenges to urban security and management. In this perspective, automated crowd analysis methods, which typically address crowd counting and the associated crowd density estimation task, have attracted increasing attention for their many potential applications [1], [2]. These include the prevention of crowd-induced disasters, such as stampedes, but also other less critical objectives such as better crowd management at public events and the design of public spaces and virtual environments.

Among the various strategies that can be followed to accomplish such tasks, a cost-effective way to perform automated crowd analysis is the use of unmanned aerial vehicles (UAVs), more commonly known as drones. Indeed, once equipped with inexpensive but powerful enough cameras and GPUs, drones can become flying computer vision devices that can be rapidly deployed for a wide range of applications, including crowd analysis for public safety [3]. However, while these perspectives are fascinating, there are also some drawbacks to be aware of. On the one hand, computer vision algorithms applied to aerial images are burdened with further difficulties, because the issues of scale and viewpoint are taken to the extreme. On the other hand, the methods commonly applied in this field, which are sophisticated and computationally intensive, do not meet the stringent real-time requirements imposed by the UAV. In other words, lightweight models that offer a good compromise between effectiveness and efficiency are vital [4].

Crowd analysis with drones has attracted attention in recent years [5]. However, despite significant progress, the proposed methods still have room for improvement to address the challenges posed by drones. Furthermore, the development and evaluation of these algorithms is hampered by the scarcity of large-scale benchmarks available, due to the difficulties in collecting and annotating videos taken by drones. In this paper, we want to contribute to this research effort by taking a step further: instead of considering crowd counting and density estimation in still frames from drones, we aim to detect crowd flow. This poses a new challenge as the goal is not only to estimate the number of people in a single high-altitude scene but also to determine how the crowd flows as a function of time. This is different from people tracking, where the goal is to track a single person or groups of people, and can lead to useful systems, as it can allow for crowd behavior analysis for better logistics and disaster prevention [6].

To this end, we propose a method for crowd flow detection from drones that works in two steps. A Fully Convolutional Network (FCN) is used to predict and generate in real-time density maps from the actual scenes captured by the camera, to locate the concentrations of people in each scene. This architecture is preferred over some other choices, as it allows for shorter inference times. The density maps obtained in the first phase then become the input for a clustering module whose goal is to recognize the groups of people within the crowd. These groups are identified by the centroids of the detected clusters, which are finally used to track the trajectories of the identified groups, following their movement during the shooting of the drone. The method was tested on the recently proposed VisDrone Crowd Counting dataset [7] used for the homonymous international challenge. The particularity of this dataset is that it is not characterized by still images but actually by frames of video sequences which are used here to perform the crowd flow detection task.

It is worth noting that, as briefly introduced, we compute an inter-frame difference between centroids to determine the

This work was supported by the Italian Ministry of University and Research within the "RPASInAir" project under grant PON ARS01_00820.

[1]Code: https://github.com/gvessio/uav-crowd-flow-detection.

crowd flow, but we do not study this flow from the point of view of physical principles such as those of fluid dynamics. In other words, what we actually do is "inter-frame density clustering". However, because this approach allows us to detect the movement of the crowd frame by frame, we use the expression "flow detection" for simplicity of terminology.

The rest of this paper is structured as follows. Section 2 reviews related work. Section 3 presents the proposed method. Section 4 describes the experimental setup and discusses the results obtained. Section 5 concludes the paper and highlights future developments of our research.

II. Related Work

There is a large body of knowledge on crowd counting and crowd density estimation in computer vision, but the dominant approach today is that based on density estimation. Early work usually applied a person or head detector via a sliding window on the image, but, although current implementations may be based on state-of-the-art object detectors such as YOLO [8], [9], these approaches still provide unsatisfactory results when asked to detect small objects in very dense crowds. To alleviate this problem, regression-based methods have been introduced that directly learn the mapping from an image to the global people count [1]. However, although the use of these methods makes the approach independent of the precise position of individuals in the crowd, which is very complex, they ignore spatial information that can be very useful for prediction. To avoid the difficulty of accurately detecting and locating people in the scene, while using spatial information, the recent trend is to learn density maps, thus incorporating the spatial information directly into the learning process [10]. Successful solutions include methods that work first at the patch level and then fuse local features [11], methods that integrate attention mechanisms [12], and cascade approaches that jointly learn people count and density maps [13].

However, while effective, these approaches are generally computationally demanding and do not meet the stringent requirements typically imposed by the UAV (limited battery, need for real-time responses). How to fine-tune deep neural architectures to achieve an optimal balance between precision and performance is an active research area. The VisDrone Crowd Counting challenge was introduced to encourage research in this direction [7], [14]; however, the solutions proposed by the participants in the challenge are not always focused on efficiency but effectiveness, as the goal is only to obtain a low people counting error. The lowest error obtained was achieved with TransCrowd [15], based on the increasingly popular Visual Transformer [16]. However, the proposed method only regresses the people count, not providing density maps that would be useful for detecting crowd flow; furthermore, transformer-based solutions are known to be computationally expensive.

A promising way to address these issues is to use FCN models. Since they do not rely on fully connected layers at all, which are actually the most expensive part of a neural network processing, they are a candidate solution for finding an accurate model, without damaging the inference time. An FCN model for aerial drone imaging is the one presented in [4], and a similar solution was also proposed by us in a previous work [17]. However, both methods were aimed at crowd detection, i.e. discriminating between crowded and uncrowded scenes; furthermore, they only provide coarse density maps, as the models have not been trained on people labels. An FCN model that has been used successfully for crowd density estimation is MobileCount [18]; however, it has never been tested on aerial imagery, so its ability to generalize beyond traditional scenes has yet to be proven. In light of these observations, in this work we leverage MobileCount, using it as a backbone network to estimate density maps, and then rely on its output to subsequently perform crowd flow detection.

Human tracking methods based on RGB cameras or other sensors that use clustering or classification models to track motion have been investigated, e.g. [19]–[21]. However, they are designed to work indoors or by involving few people from a frontal perspective. The work most linked to ours, which takes into account drone-captured images, is [22] where the same authors who propose the VisDrone Crowd Counting dataset present a model that jointly solves density map estimation, localization, and tracking. This model differs from ours in that it uses a complex and expensive pipeline aimed at tracking individual trajectories. As far as we know, there is no work in the literature addressing crowd flow detection in drone videos, which pose significantly different challenges than traditional settings. The goal of this paper is to fill this gap; more specifically, we aim to trace the centroids identifying groups of people by exploiting the spatial information learned and expressed through the density maps.

III. Proposed Method

As schematized in Fig. 1, the proposed method for crowd flow detection works mainly in two steps. An FCN is used to estimate a density map from each frame of the video sequence shot by the drone. The dense areas in these density maps are then clustered to identify groups of people within them, not necessarily following the same direction. Finally, the displacements of the centroids detected between the frames are calculated, so that their direction of movement can be identified. The idea of using a density estimation method in combination with clustering algorithms, instead of tracking the movement of each individual, is motivated by the complexity and computational expense of this strategy. Furthermore, individual tracking may be not only impractical but also inessential, as in crowd management scenarios it is important to recognize the overall flow of people rather than the precise location of each individual person in the scene. A density estimation method will focus only on high-density areas, i.e. those corresponding to the concentrations of people, and will be inherently robust to occlusion, which would heavily affect a people detector, especially from a high altitude.

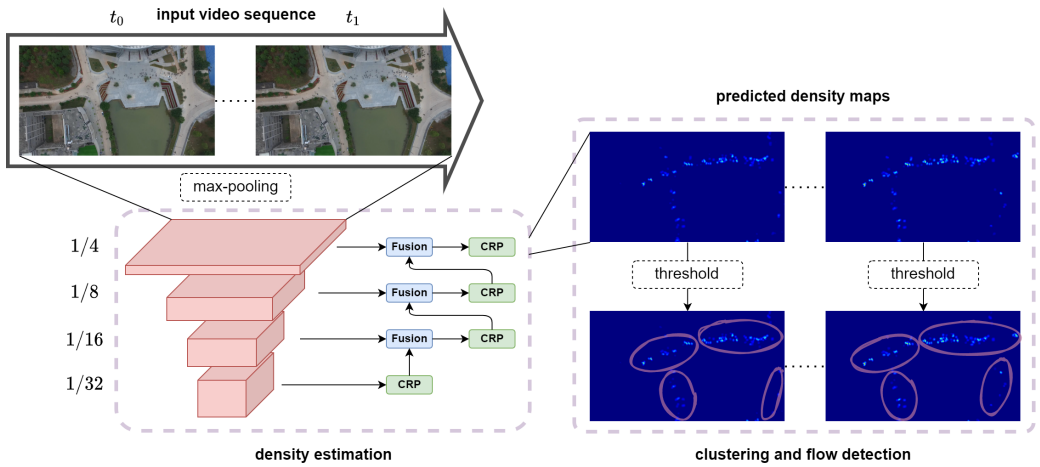


Fig. 1: Schema of the proposed method. Consecutive frames taken by a drone at time t_0 and t_1 (with $t_1 > t_0$ and separated by any time interval) are fed into a MobileCount-based density estimation module. The predicted density maps are then thresholded and provided as input to the clustering module. The groups of people within the crowd are identified by their centroids and finally the shift of the centroids from t_0 to t_1 is used to determine their direction of movement.

A. *Density Estimation*

In this section, we describe the MobileCount model for crowd density estimation [18]. It was chosen because it is one of the most efficient lightweight models that offers a good compromise between precision and performance. MobileCount is an FCN encoder-decoder that can be trained end-to-end. The popular MobileNetV2 [23] is used as the encoder, which is suitable for mobile and embedded applications. MobileNetV2 is based on an inverted residual structure where the residual block input and output are thin bottleneck layers as opposed to traditional residual models that use expanded representations in the input. Furthermore, lightweight depth-wise convolutions are used to filter features in the intermediate expansion layer. To further save on computation, MobileCount uses only 4 bottleneck layers; moreover, to make the model even lighter, the input resolution is reduced at the beginning of the encoder by adding a 3×3 pooling with a stride of 2.

As for the decoder, the correct estimate of the crowd density of a particular pixel cannot be based only on the information of the pixel itself, so it is necessary to incorporate context information of multiple scales to address the different sizes of people. For this purpose, the Light-Weight RefineNet decoder [24] originally designed for semantic segmentation is used. Light-Weight RefineNet is a lighter version of RefineNet [25] which can in principle be combined with any backbone encoder. The decoder propagates the last output from the backbone with the lowest resolution through a chained residual pooling (CRP) block and then feeds it into a fusion block along with the penultimate feature map. Within the fusion block, each path is convolved with 1×1 convolution, and the low-resolution feature maps are up-sampled at a high resolution between the paths. Two paths are then added together and further propagated through multiple CRP and fusion blocks until the desired resolution is achieved. For the sake of brevity, the details on these blocks are omitted and can be found in the original papers [24], [25].

Finally, the generated feature maps are fed into the prediction layer to produce the final density map. This layer applies a 1×1 convolution, embedding the feature vector at each pixel of an input feature map to a density value. The resulting density map is up-sampled to the original image size by bilinear interpolation to match the original size. As a loss function, the overall architecture is trained to minimize the mean squared error between the predicted and the ground truth density map by backpropagation:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \left\| D^P(i) - D^{GT}(i) \right\|_2^2$$

where N is the number of samples, $D^P(i)$ and $D^{GT}(i)$ are the predicted and ground truth density map, respectively, and $\|\cdot\|_2$ is the Euclidean distance.

B. *Crowd Flow Detection*

Since our goal is to determine the direction of movement of the crowd, and a crowd can be seen as multiple groups of people not necessarily following the same direction, our main idea is to cluster the dense areas in the density maps obtained from MobileCount applied to the video sequence shot by the drone. This allows us to identify groups of people based on their centroids. Then, we can recalculate the centroids in each frame of the entire video sequence allowing us to track their shift frame by frame.

More precisely, to calculate the actual displacement of the recognized groups of people, and to determine the direction of their movements, we calculate the coordinate difference of the centroids calculated in different frames. In other words, the displacement of the centroids in the current frame at time t is calculated with respect to the centroids detected in the

frame at time $t-1$, and so on for the entire frame group. Since we are in two dimensions, this displacement is simply calculated using the Euclidean distance between the (x, y) coordinates. The main assumption of a crowd flow detector would be that a given group of people, i.e. a centroid in this case, can potentially move between successive frames, but its distance from its position in the previous frame would be less than the distance from all other centroids in the current frame. Therefore, to determine if we can associate the newly calculated centroids to the existing ones, and thus in which direction they have moved, we calculate the Euclidean distance between each pair of centroids in each frame and keep the minimum distances between the pairs to match them. In this way, we can classify each shift as facing one of the four typical cardinal points, namely North, South, East, and West, plus the intermediate points North-East, North-West, South-East, and South-West. The possibility of no movement is also considered. The overall displacement can then be obtained in several ways, for example by simply comparing two frames separated by a previously set time interval (less expensive) or by calculating a "moving average" over an entire sequence (much more expensive). It is worth noting that, in subsequent frames, new people may appear in the scene. In this case, new centroids will be detected and their movement will be traced in the following frames. Conversely, it can also happen that some people have left the field of view. In this case, the old centroids will not match the current centroids and therefore will be discarded.

Since it is not possible to know in advance the number of clusters in the crowd, we have experimented with clustering methods that do not require pre-specification of the number of clusters. There are several algorithms available; among these, we have experimented with three popular choices:

- Hierarchical clustering (henceforth Hclustering): classic hierarchical clustering that uses a bottom-up approach where each observation starts in its own cluster and the clusters are subsequently merged [26]. The single linkage criterion is used as a merging strategy, which minimizes the distance between the closest observations of cluster pairs. Finally, a cut-off threshold is used to form clusters.
- Mean Shift: this is a centroid-based algorithm, which works by updating the candidates for the centroids as an average of the points within a given region. These candidates are then filtered in a post-processing step to eliminate near-duplicates [27].
- HDBSCAN: this is an extension of the popular DBSCAN [28] which converts the more classic method into a hierarchical clustering algorithm, and then uses a technique to choose clusters based on their stability. The algorithm was designed to be as fast as possible and robust to parameter selection [29].

Finally, it is worth noting that, although accurate, predicted density maps are characterized by non-normalized values and may contain noise. To make the density maps more suitable for clustering while focusing only on the most relevant information, we previously threshold (with an empirically chosen threshold τ) the pixel values, so that any value below the threshold is considered as a background. Furthermore, we apply a min-max normalization to limit their range in $[0, 1]$.

C. Ground Truth Generation

We distinguish two types of ground truth: one for the density estimation task, and the other for its clustering. The first ground truth consists in the generation of density maps related to images that are assumed to be annotated with the coordinates of the position of the head of the persons depicted. In other words, a real image with labeled people $\mathbf{x}$ needs to be converted into a crowd density map $D(\mathbf{x})$. Following the seminal paper by Zhang et al. [30], this is done by convolving a delta function δ with a Gaussian kernel:

$$D(\mathbf{x}) = \sum_{i=1}^{M} \delta(\mathbf{x} - \mathbf{x}_i) * G_{\sigma_i}(\mathbf{x})$$

where $\mathbf{x}_i$ $(i = 1, \ldots, M)$ is the coordinate pair of the i-th head; δ equals 1 when $\mathbf{x} = \mathbf{x}_i$, 0 otherwise; and G_σ is the Gaussian kernel. As in [30], this kernel is adaptive to the local geometry around each data point, since the standard deviation is $\sigma_i = \beta \bar{d}^i$, where $\beta = 0.3$ is found empirically and $\bar{d}^i$ is defined as the average distance to the nearest neighbors.

In addition, since there is no label in VisDrone regarding the positioning of centroids within the original images, we followed a simple strategy to evaluate the effectiveness of the proposed crowd flow detector. This was based on the application of the same clustering algorithms tested on the ground truth density maps described above. This allows us to evaluate the correctness of the centroids identified within the density maps provided by MobileCount.

IV. Experiment

In the following subsections, we describe the dataset used to evaluate the effectiveness of the proposed method, the performance metrics considered and finally we discuss the results obtained.

A. Dataset

The literature landscape is not populated with datasets for crowd counting and density estimation from drones with video sequences captured by optical cameras. The dataset that best suited our purposes for evaluating crowd flow detection was the VisDrone Crowd Counting dataset [7], [14]. This benchmark consists of 112 sequences with the resolution of 1920×1080, including 82 video sequences for training ($2,420$ frames in total) and 30 sequences for testing (900 frames in total). Images were acquired by various cameras mounted on drones to maintain diversity, for 70 different scenarios in four different cities in China. To avoid overfitting to particular scenes, the images were collected in the training and test subset in different but similar locations. People were annotated manually with dots in each video frame.

Compared to benchmark datasets focused on crowd counting in surveillance scenes, VisDrone presents some challenges

due to the scenes captured by drones. Object scales are extremely small due to the high shooting altitude of drones. Crowds are sparse across the video frames, as each crowd can hold a few to dozen people. Finally, in different sequences the crowds are surrounded by very different backgrounds.

B. Performance Metrics

To evaluate the accuracy of the density map estimation task, we calculated generally agreed "image-level" metrics. The mean absolute error is the expected value of the absolute error, thus determining the accuracy of the estimates; it is defined as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|$$

where $\hat{y}_i$ is the crowd count obtained by adding all non-background pixels in the i-th density map, while y_i is the corresponding ground truth crowd count. Similarly, the root mean squared error is the square root of the mean of the squared differences between the predicted and the actual count, thus penalizing larger errors more; it is calculated as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

Both metrics are negatively oriented, which means lower values are better.

To also evaluate the effectiveness of the clustering algorithms we tested, we used popular metrics. The silhouette coefficient is a measure of how similar a data point is to its cluster compared to other clusters [31]. The score for each sample is given by:

$$SC = \frac{b-a}{\max(a,b)}$$

where a is the average distance between a sample and all other points of the same cluster, while b is the average distance between a sample and all other points in the nearest cluster. The overall coefficient is given as the average of the scores for each sample. The coefficient ranges from -1 to $+1$, where a high value indicates that data points are well matched to their cluster and poorly matched to neighboring clusters. The Davies-Bouldin index indicates the average "similarity" between clusters, where similarity is a measure that compares the distance between clusters with the size of the clusters [32]. It is defined as:

$$DB = \frac{1}{k}\sum_{i=1}^{k}\max_{i\neq j} R_{ij}$$

with k the number of clusters and $R_{ij} = \frac{s_i+s_j}{d_{ij}}$, where s_i is the average distance between each point of cluster i and the centroid of that cluster, and d_{ij} is the distance between the centroids of cluster i and j. Zero is the lowest possible DB score: values closer to zero indicate a better clustering.

Both the silhouette coefficient and the Davies-Bouldin index are measures of internal validity that assume that the ground truth labels are not known. However, applying already known external measures is not feasible in our density-based context as they assume an exact match between discretized category labels. Since we are interested in measuring how correctly the detected centroids represent relevant groups of people within the scene, we propose here a new ad-hoc metric, which we have called Mean Coordinate Matching Error (MCME). Basically, the metric measures the average distances between the predicted centroids and the ground truth centroids, which were obtained as described above by running the clustering algorithms on the ground truth density maps. For a single frame, let:

$$A = \left\{ \left(C^P, C^{GT}_{\min}\right) \;\middle|\; \begin{array}{l} C^P \in P, \\ C^{GT}_{\min} = \arg\min_{C^{GT}\in GT} \left\|C^P - C^{GT}\right\|_2 \end{array} \right\}$$

$$B = \left\{ \left(C^P_{\min}, C^{GT}\right) \;\middle|\; \begin{array}{l} C^{GT} \in GT, \\ C^P_{\min} = \arg\min_{C^P\in P} \left\|C^P - C^{GT}\right\|_2 \end{array} \right\}$$

where $GT = \{C^{GT}_1, C^{GT}_2, \ldots\}$ and $P = \{C^P_1, C^P_2, \ldots\}$ are the ground truth and predicted centroids, respectively. Then:

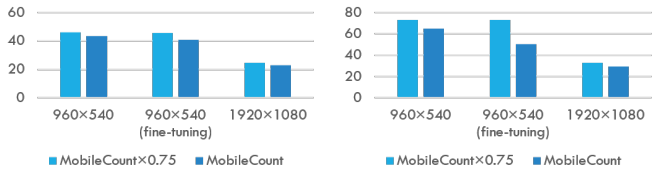
$$MCME = \frac{1}{|A \cup B|}\sum_{(C^P, C^{GT})\in A\cup B}\left\|C^P - C^{GT}\right\|_2$$

Each centroid of a set (say P) is associated with the nearest neighbor centroid of the other set (GT) since both sets of centroids are assumed to represent the same structure in the data. This association must be symmetrical, i.e. from P to GT and vice versa, because a centroid in P can represent part of a cluster in GT that has been split into two clusters in P, but it can also represent a cluster in P that has merged two clusters in GT. The overall score on a video sequence can be obtained by averaging the individual scores. The proposed metric aims to "punish" the model both when the predictions are very far from reality, and when actual groups are not identified or groups that do not exist are identified. Instead, lower values for MCME will indicate that the predicted centroids match the ground truth centroids, i.e. they are the same in number and are very close.

C. Results

The proposed method has been implemented in Python, using the `PyTorch` library for MobileCount, `scipy` for Hclustering, and `scikit-learn` for Mean Shift and HDBSCAN. MobileCount has been trained with an early stopping strategy on a hold-out validation set (58 epochs to reach the best model), using Adam as an optimizer with a learning rate of 10^{-3}, on Google Colab Pro. The overall inference, mainly consisting of predicting a density map and clustering it for each frame, was tested on the same platform. Finally, as the cut-off threshold for the Hclustering algorithm, a value of 80 pixels was empirically chosen.

In the following, we report the results of two families of experiments related to the two tasks of density estimation and clustering for crowd flow detection.



(a) MAE (b) RMSE

Fig. 2: Crowd counting performance (*fine-tuning* stands for the double fine-tuning from the Object Detection dataset).

1) Density Estimation: As shown in Fig. 2, we experimented with three different configurations:

- First, we scaled the images in the VisDrone Crowd Counting dataset to 960×540 to try to reduce the inference time.
- In a second configuration, we experimented with a *double* fine-tuning approach where the weights of the MobileNetV2 encoder, already pre-trained on ImageNet, were fine-tuned on the VisDrone Object Detection dataset [33] (used for a different task challenge), and then fine-tuned on the VisDrone Crowd Counting dataset. This solution was intended to try to inject more knowledge of aerial imagery into the model. Since VisDrone Object Detection has images with lower resolution than VisDrone Crowd Counting, we have not up-scaled them to avoid a loss of quality.
- Finally, the third configuration concerns training only on VisDrone Crowd Counting but directly on the original images of size 1920×1080.

In addition, for each experiment, we also used a version of MobileCount in which the number of density maps in each layer, i.e. its depth, was reduced by a factor of $\times 0.75$, to make the model even lighter.

From the figure, it can be seen that although the double fine-tuning strategy slightly improved the results obtained without using it, the experiments where the image size was reduced both show significantly worse performance than the scenario where the original image size is retained. This can be easily explained considering that crowd counting and density estimation from a relatively high altitude are highly dependent on input resolution: even a non-drastic reduction—960×540 is quite high compared to other computer vision tasks—dramatically worsens the accuracy of the results. As for the difference between the two versions of MobileCount, the lighter did not have an appreciable impact on computational savings, so all in all we decided to keep the original version in the subsequent experiments. The best results are an MAE of 22.99 and an RMSE of 29.64. Examples of input images and corresponding density maps are shown in Fig. 3.

As mentioned above, the current state-of-the-art is held by TransCrowd [15], with an MAE of 9.59 and an RMSE of 15.57. However, this transformer-based method does not provide density maps, which are actually needed to detect the flow of people. Furthermore, although the frame rate is not explicitly mentioned in the paper, transformer-based methods are known to be still computationally demanding.

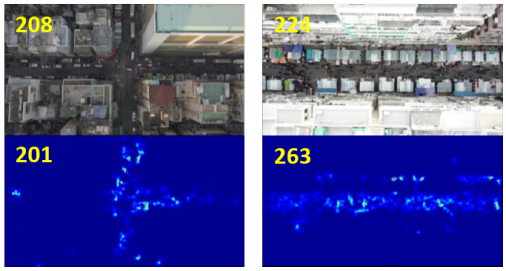
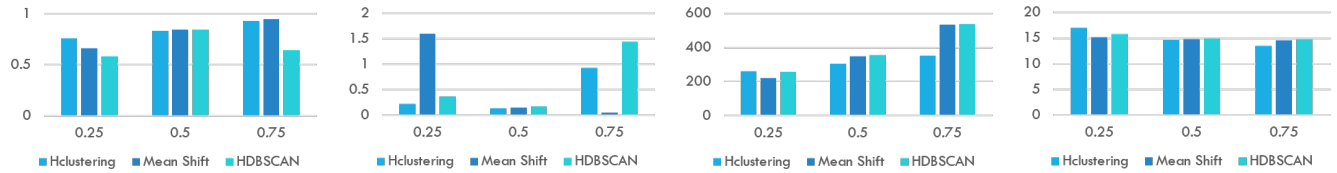


Fig. 3: Examples of density maps, and predicted count and corresponding ground truth. Although large counting errors can be made as in the case on the right, the method is still effective in detecting areas of people concentration.

2) Crowd Flow Detection: To assess the accuracy of crowd clustering, several experiments were performed by combining one of the three clustering algorithms we tested, namely Hclustering, Mean Shift and HDBSCAN, and three different values for the threshold τ (which was used to filter the predicted density maps), namely 0.25, 0.50 and 0.75. A higher threshold essentially maintains the areas where the model was more confident about the presence of people in those areas.

Figure 4 shows the obtained results averaged over all test sequences. Regarding the silhouette coefficient, it can be seen that all algorithms perform well with Hclustering showing the best results on average. As for the other internal metric, that is the Davies-Bouldin index, the performance is more variable as the threshold varies, with the best results for all the algorithms obtained with a threshold of 0.50. As for the newly proposed MCME metric, the three clustering algorithms show quite comparable results with decreasing performance as the threshold increases, denoting a proportionality between these two values. One possible explanation is that a severe threshold moves the predicted centroids away from the actual centroids. The best results, slightly above 200, are obtained from the three algorithms with a threshold of 0.25. Considering that full resolution images were used to achieve these results, an average error of 200 pixels seems not so large.

Finally, regarding the inference time, it should be noted that it includes all image processing, from estimating the density map to filtering and normalization, and calculating clustering and displacement. Since the overall inference time is still quite high, we note that crowd flow detection should be performed not for every pair of subsequent frames, but after a certain amount of time has elapsed. After all, given the high frame rate of the camera, the movement of groups of people between frames is minimal and in some cases zero. For this reason, calculating the movement of a group of people with such precision is excessively expensive from a computational point of view and does not correspond to a real advantage.



(a) Silhouette coefficient (b) Davies-Bouldin index (c) MCME (d) Inference time [s]

Fig. 4: Clustering performance.

V. Conclusion

In this paper, we have addressed the problem of crowd flow detection from drones which was still an unexplored research direction. Having such a system can be useful for several security and management applications, especially in smart city scenarios. In particular, the joint exploitation of crowd density estimation within a video sequence shot by a drone together with clustering techniques has provided promising results. It is worth noting that the use of different background thresholds to filter the density map values can be set according to the specific application to meet different tasks. For example, the safe landing of drones would require a less stringent threshold, as the risk of running into false negatives would be detrimental to safety concerns. Conversely, in video surveillance scenarios, a stricter threshold can be used to promote better accuracy.

A future development of the research presented in this paper could be to investigate the possibility of integrating the clustering task directly into a multi-task deep learning model trained to perform density estimation and clustering simultaneously. Such a strategy can help significantly reduce inference time, which still needs to be improved. Finally, another future work concerns the experimentation of the model on new real situations aboard a drone to best calibrate the parameters considered. Testing the generalizability of the method to different urban and non-urban contexts can increase confidence in UAV technology.

References

[1] V. A. Sindagi and V. M. Patel, "A survey of recent advances in CNN-based single image crowd counting and density estimation," *Pattern Recognition Letters*, vol. 107, pp. 3–16, 2018.

[2] B. Li, H. Huang, A. Zhang, P. Liu, and C. Liu, "Approaches on crowd counting and density estimation: a review," *Pattern Analysis and Applications*, pp. 1–22, 2021.

[3] Y. Akbari, N. Almaadeed, S. Al-maadeed, and O. Elharrouss, "Applications, databases and open computer vision research from drone videos and images: a survey," *Artificial Intelligence Review*, vol. 54, no. 5, pp. 3887–3938, 2021.

[4] M. Tzelepi and A. Tefas, "Graph embedded convolutional neural networks in human crowd detection for drone flight safety," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2019.

[5] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, "Detection and tracking meet drones challenge," 2021.

[6] V. J. Kok, M. K. Lim, and C. S. Chan, "Crowd behavior analysis: A review where physics meets biology," *Neurocomputing*, vol. 177, pp. 342–362, 2016.

[7] D. Du, L. Wen, P. Zhu, H. Fan, Q. Hu, H. Ling, M. Shah, J. Pan, A. Al-Ali, A. Mohamed, B. Imene, B. Dong, B. Zhang, B. H. Nesma, C. Xu, C. Duan, C. Castiello, C. Mencar, D. Liang, F. Krüger, G. Vessio, G. Castellano, J. Wang, J. Gao, K. Abualsaud, L. Ding, L. Zhao, M. Cianciotta, M. Saqib, N. Almaadeed, O. Elharrouss, P. Lyu, Q. Wang, S. Liu, S. Qiu, S. Pan, S. Al-Maadeed, S. D. Khan, T. Khattab, T. Han, T. Golda, W. Xu, X. Bai, X. Xu, X. Li, Y. Zhao, Y. Tian, Y. Lin, Y. Xu, Y. Yao, Z. Xu, Z. Zhao, Z. Luo, Z. Wei, and Z. Zhao, "VisDrone-CC2020: The vision meets drone crowd counting challenge results," in *Computer Vision – ECCV 2020 Workshops*, A. Bartoli and A. Fusiello, Eds. Cham: Springer International Publishing, 2020, pp. 675–691.

[8] W. Lan, J. Dang, Y. Wang, and S. Wang, "Pedestrian detection based on yolo network model," in *2018 IEEE international conference on mechatronics and automation (ICMA)*. IEEE, 2018, pp. 1547–1551.

[9] V. Molchanov, B. Vishnyakov, Y. Vizilter, O. Vishnyakova, and V. Knyaz, "Pedestrian detection in video surveillance using fully convolutional YOLO neural network," in *Automated visual inspection and machine vision II*, vol. 10334. International Society for Optics and Photonics, 2017, p. 103340Q.

[10] G. Gao, J. Gao, Q. Liu, Q. Wang, and Y. Wang, "CNN-based density estimation and crowd counting: A survey," *arXiv preprint arXiv:2003.12783*, 2020.

[11] L. Zhu, C. Li, Z. Yang, K. Yuan, and S. Wang, "Crowd density estimation based on classification activation map and patch density level," *Neural Computing and Applications*, vol. 32, no. 9, pp. 5105–5116, 2020.

[12] G. Zhang, Y. Pan, L. Zhang, and R. L. K. Tiong, "Cross-scale generative adversarial network for crowd density estimation from images," *Engineering Applications of Artificial Intelligence*, vol. 94, p. 103777, 2020.

[13] V. A. Sindagi and V. M. Patel, "CNN-based cascaded multi-task learning of high-level prior and density estimation for crowd counting," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2017, pp. 1–6.

[14] Z. Liu, Z. He, L. Wang, W. Wang, Y. Yuan, D. Zhang, J. Zhang, P. Zhu, L. Van Gool, J. Han *et al.*, "VisDrone-CC2021: The vision meets drone crowd counting challenge results," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2830–2838.

[15] D. Liang, X. Chen, W. Xu, Y. Zhou, and X. Bai, "TransCrowd: Weakly-supervised crowd counting with transformer," *arXiv preprint arXiv:2104.09116*, 2021.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[17] G. Castellano, C. Castiello, C. Mencar, and G. Vessio, "Crowd detection in aerial images using spatial graphs and fully-convolutional neural networks," *IEEE Access*, vol. 8, pp. 64 534–64 544, 2020.

[18] P. Wang, C. Gao, Y. Wang, H. Li, and Y. Gao, "MobileCount: An efficient encoder-decoder framework for real-time crowd counting," *Neurocomputing*, vol. 407, pp. 292–299, 2020.

[19] V. Gajjar, A. Gurnani, and Y. Khandhediya, "Human detection and tracking for video surveillance: A cognitive science approach," in *Proceedings of the IEEE international conference on computer vision workshops*, 2017, pp. 2805–2809.

[20] Y. Xiao, V. R. Kamat, and C. C. Menassa, "Human tracking from single RGB-D camera using online learning," *Image and Vision Computing*, vol. 88, pp. 67–75, 2019.

[21] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for 3D LiDAR-based human detection: Experimental analysis of point cloud clustering and classification methods," *Autonomous Robots*, vol. 44, no. 2, pp. 147–164, 2020.

[22] L. Wen, D. Du, P. Zhu, Q. Hu, Q. Wang, L. Bo, and S. Lyu, "Detection, tracking, and counting meets drones in crowds: A benchmark," in

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 7812–7821.

[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[24] V. Nekrasov, C. Shen, and I. Reid, "Light-Weight RefineNet for real-time semantic segmentation," *arXiv preprint arXiv:1810.03272*, 2018.

[25] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1925–1934.

[26] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview, ii," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 7, no. 6, p. e1219, 2017.

[27] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[28] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 3, pp. 1–21, 2017.

[29] L. McInnes, J. Healy, and S. Astels, "HDBSCAN: Hierarchical density based clustering," *Journal of Open Source Software*, vol. 2, no. 11, p. 205, 2017.

[30] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 589–597.

[31] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[32] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 224–227, 1979.

[33] P. Zhu, L. Wen, D. Du, X. Bian, H. Ling, Q. Hu, Q. Nie, H. Cheng, C. Liu, X. Liu *et al.*, "VisDrone-DET2018: The vision meets drone object detection in image challenge results," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.