

Leveraging On-board UAV Motion Estimation for Lightweight Macroscopic Crowd Identification

Khizar Anjum⁺, Tahmeed Chowdhury⁺, Sreeram Mandava⁺, Benedetto Piccoli^{*}, and Dario Pompili⁺

⁺Dept. of Electrical and Computer Engineering, ^{*}Dept. of of Mathematical Sciences

Rutgers University, NJ, USA

{khizar.anjum, tsc91, sm2244, pompili}@rutgers.edu, piccoli@camden.rutgers.edu

Abstract—This paper introduces a novel, lightweight, on-board approach to crowd pattern identification, ingeniously using the processes of existing video compression standards, particularly H.264 or MPEG-4. Piggy-backing on the H.264 video-encoding algorithm, we propose real-time crowd pattern recognition and identification methodologies that can identify macroscopic patterns in as low as 2 milliseconds on NVIDIA TX2, resulting in around 45× execution time reduction compared to existing approaches. Furthermore, we introduce a temporally aware approach to pinpoint and adapt to crowd movement patterns, continuously recalibrating as a drone’s Point Of View (POV) varies or observed motions diverge. Evaluating our method against publicly available datasets, we emphasize our system’s performance and computational advantages, especially when faced with real-time observational shifts. In conclusion, our approach elegantly bridges the gap between crowd safety imperatives and the challenges of UAV monitoring, heralding a new era of real-time drone-centric crowd management intelligence.

Index Terms—Crowd Pattern Recognition, Crowd Surveillance, On-board Computation, Video Encoding

I. INTRODUCTION

In recent years, Unmanned Aerial Vehicles (UAVs) have transitioned from niche gadgets to ubiquitous tools, finding use in diverse fields such as agriculture, cinematography, and public safety [1]. Their unique aerial perspective offers an unparalleled advantage, particularly regarding monitoring scenarios like mass gatherings, protests, or public events. UAVs, also known as drones, can both observe and guide the crowd in the event of an emergency. The challenges unique to drones include unreliable connectivity—in contrast to wired fixed cameras—limited on-board resources, and complex coordination, while potential advantages include mobility, adaptability, and distinct Points Of View (POVs). Given these challenges and opportunities, a traditional centrally located control room-based monitoring approach becomes untenable and unscalable for drones [2].

Therefore, it is crucial to rethink monitoring approaches to use the mobility of these new tools to its fullest. Therefore, a hierarchical management approach is needed in which each drone has some real-time capability on board for local situational awareness, and there exists a central server that can deploy and direct drones to certain Regions Of Interest (ROIs), and issue directives. This approach (as shown in Fig. 1) is well suited for managing mobile crowds, whose intent and direction could change depending on time, location, and type of event [3].

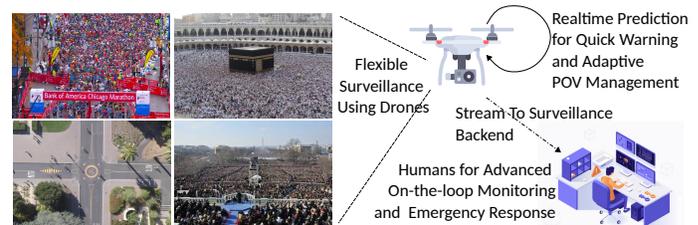


Fig. 1: Our concept of drone-based crowd management: Autonomous drones can predict crowds in real-time, focus more on potentially anomalous areas, and issue local warnings. A central server, on the other hand, is involved in global monitoring as well as coordination.

To realize such a management approach, however, several innovations are needed, both in terms of algorithmic solutions and physical infrastructure. Due to the size of this problem, it is impossible to tackle it as a whole in one technical paper. As a result, we limit the scope of this paper to a novel algorithmic approach to crowd motion pattern recognition and identification that can be extended to crowd behavior recognition, management, and overcrowding prevention while keeping in mind that it is a piece of the overall puzzle directed toward rapid, mobile, and adaptive crowd monitoring in the presence of unreliable connectivity. Recognizing the gap in available and needed resources in a drone with a limited battery, a compelling need for the algorithm to be lightweight emerges, as existing algorithms could put undue pressure on the device and jeopardize public safety. Therefore, our proposal can help improve safety by minimizing stress while providing local decision-making capabilities.

Our Approach: Achieving real-time decision-making on resource-constrained platforms such as drones is challenging, so it has given impetus to a growing body of research that focuses on offloading computation to nearby powerful nodes, strategically placed in vehicles close by or connected to the grid, known as Mobile Edge Computing (MEC) [4]. While an offloading approach might be faster than local processing in some scenarios, offloading during crowd monitoring involves competition with requests from users in the crowd itself, thwarting real-time decision-making. Implementing an exclusive communication infrastructure for drones, on the other hand, is not only not scalable, but also limits the mobility of drones—one of the key advantages of drone usage. With these challenges in mind, we present a lightweight approach for on-

board real-time crowd pattern recognition and identification, taking advantage of video encoding processes (typically H.264 or MPEG-4 [5]) already running as the drone records video.

Our Contributions: We make the following contributions:

- We introduce a novel algorithm for macroscopic crowd pattern detection and identification that shares the internal motion estimation done for video recording.
- We propose a temporally aware approach to identify macroscopic crowd movement that adapts its output as the drone changes its POV, or as the observed motion changes in the current POV.
- We evaluate our method with publicly available datasets, showcasing our performance and computational savings.

Paper Outline: The remainder of the paper is organized as follows. In Sect. II, we position our work with respect to the related literature. In Sect. III, we explain our proposed resource-sharing approach. In Sect. IV, we evaluate our approach on a variety of light and dense crowd datasets. Finally, in Sect. V, we conclude and discuss future work.

II. RELATED WORK

This work brings computational awareness to the field of crowd identification and monitoring and therefore has intersecting ideas from both research bodies. Consequently, the review is divided into multiple subparts as follows.

UAV-based crowd monitoring: Since the inception of UAVs, experts have strived to harness their natural mobility to monitor crowds. De Moraes and De Freitas [6] elaborated on a system that employs multiple UAVs for this purpose. However, their study focuses primarily on drone coordination. In contrast, our study emphasizes the goal of transforming real-time video footage into actionable insights into crowd dynamics. Other works, such as, Rodriguez-Canosa et al. [7] suggest a technique for real-time detection and tracking of objects via UAV cameras, but it underperforms with densely populated crowds.

Macroscopic Crowd Pattern Recognition: There has been active work on macroscopic crowd patterns. Matkovic et al. [8] use Brox Flow [9] as a starting point and build on it to achieve simple identification and anomaly detection in crowds on a macroscopic scale. However, the approach is complex, hindering its successful implementation in resource-constrained devices, such as drones. Similarly, Zhang et al. [10] use a trajectory tracking approach for crowd segmentation, but their approach is slow—on the order of a few seconds per frame. Finally, Almeida and Jung [11]—using Farneback Flow [12]—propose a relatively lightweight approach for crowd pattern recognition, but their approach is not suitable for dynamic UAVs.

Deep Learning-based Crowd Prediction: Optical flow is defined as the pixel-by-pixel motion observed in a video sequence. Methods for estimating flow range from low to high complexity, such as Fast-FlowNet [13], and Deep Matching [14]. These approaches are based on neural networks and are quite resource-hungry, taking a few seconds for a single

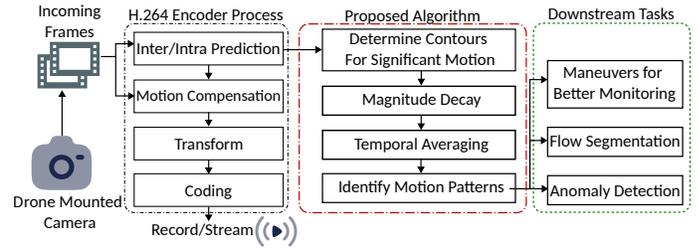


Fig. 2: Flow diagram representing the entire framework from taking incoming video frames, producing block-matching motion estimation, contouring, detecting, and identifying dominant motion patterns.

prediction on a CPU and a few hundred milliseconds on an NVIDIA GTX Titan GPU [15], and therefore are untenable for a small form factor drone. Fast-FlowNet [13] reduces the computation time to around 500 ms on a small NVIDIA TX2 GPU, which is a great improvement, but still not fast enough for real-time flow estimation. As a result, approaches relying on flow, such as Mehran et al. [16], are also unsound for real-time execution on-board.

Video Coding & Streaming: There has been active work on video coding has been conducted in the last decade culminating in the production of multiple standards, namely, High-Efficiency Video Coding (HEVC), Joint Exploration Model (JEM), and Versatile Video Coding (VVC) [17], to meet the growing need for video streaming across the globe. The improvement in coding efficiency has been tremendous; for example, VVC achieves a bit rate reduction of approximately 50% over HEVC, which in turn achieves a bit rate reduction of 50% over previous Advanced Video Coding (AVC) [5], all for the same video quality. This work capitalizes on these developments, choosing to piggyback on the H.264 video-encoding algorithm resulting in efficient crowd pattern identification on-board.

III. PROPOSED SOLUTION

In this section, we start by discussing the motion estimation shared by our proposed algorithm, then detail our crowd pattern detection approach, and finally end with the lightweight crowd pattern identification method based on conditional logic rules. Fig. 2 illustrates this process in a flow diagram.

A. Motion Estimation

The basic idea behind hybrid video encoding algorithms—such as H.264—is the recognition of both spatial and temporal redundancies in video format. Motion estimation and compensation are used to address temporal redundancy. Two types of motion estimation are used in H.264 encoding, namely, intra- and inter-prediction. Intra-prediction deals with encoding single frames without reference to other frames, while inter-prediction deals with motion between adjacent frames and is primarily the motion estimation we are interested in. For inter-prediction, frames are divided into smaller, equally-sized blocks called macroblocks. Usually, 16×16 macroblocks are used, but smaller 8×8 and 16×8 macroblocks could also be used. To estimate the motion, a block-matching algorithm compares the

macroblocks between two adjacent frames. These algorithms search for the closest macroblock in the subsequent frame for each macroblock in the subsequent frame, optimizing a cost function. MPEG-4 uses the Sum of Absolute Differences (SAD) as its cost function, defined as $\sum_{i=0}^{p-1} \sum_{j=0}^{q-1} |f_k(x+i, y+j) - f_{k-1}(u+i, v+j)|$, where f_k and f_{k-1} are the k -th and $(k-1)$ -th frames, respectively, p and q are the length and width of the macroblock, respectively, and $x, y, u,$ and v are indices for individual macroblocks in each frame. A search algorithm is normally used to optimize block matching, as a brute-force search can be very computationally intensive. Many search algorithms have been devised—e.g., Line Diamond Parallel Search (LDPS), Four Step Search (FSS), etc.—, and a particular implementation of the H.264 standard might use one or the other based on its constraints.

B. Macroscopic Pattern Detection

The intermediate output from H.264, in the form of motion vectors corresponding to each macroblock, is then passed to Algo. 1. We describe each step in detail as follows.

Algorithm 1: Detection and Identification Algorithm at a High Level

```

 $\gamma \leftarrow 0.8;$ 
comb, norm  $\leftarrow$  zeros_like(frame);
while video stream is active do
    SME = getSharedMotionEstimate(currentFrame);
    contours = findContours(SME);
    removeSmallContours(SME, contours);
    comb =  $\gamma \times$  comb + SME[magnitude];
    norm[magnitude] = normalize(comb);
    norm[angles] =
         $\alpha \times$  norm[angles] +  $(1 - \alpha) \times$  SME[angles];
    normContours = getContours(norm);
    removeSmallContours(norm, normContours);
    turns = calculateTurns(norm);
    identify(turns);
end

```

Contouring: To detect the pattern of crowd movement, the first step is to group contiguous motion vectors. We use contours as a means of locating contiguous motions throughout the frame, giving rise to the potential of observing multiple crowd patterns within a frame and saving computation by focusing on only the contours rather than the whole frame. It also helps to remove motion noise and aberrations caused by drone movement due to wind or obstacles, as well as imperfections in H.264 video encoding. Furthermore, to focus only on dominant motion vectors and reduce computation, we remove contours that are 1/4 or less of the size of the largest contour in the frame.

Magnitude Decay: Furthermore, as UAVs experience frequent video jitter and can change their POV, we introduce a decay factor $\gamma \in [0, 1]$. This factor deals with jitter in the

short term and with changing POV in the long term. γ reduces the intensity of the previous motion estimates with each new iteration. Decayed old estimate and new motion estimate are then added together. This leads to smooth transitions between motion patterns and removes intermittent jitter. Furthermore, as seen in Algo. 1, magnitudes of the combined magnitude estimate are normalized to a range of $[0, 255]$, and then converted to integers for faster computation.

Temporal Averaging: For motion angles, a decay approach does not make theoretical sense, which is why we use a temporal moving average approach to filter out noise in motion angles. For this purpose, an averaging ratio $\alpha \in [0, 1]$ is defined where $\alpha = 0$ means that there is no averaging, and all detection and identification are performed from the new motion estimate, and vice versa for $\alpha = 1$. The combined angle estimate is then used for further identification.

C. Macroscopic Pattern Identification

Identification is a crucial step in promoting crowd safety systems via UAVs, and to that end, we introduce a novel approach to identify dominant motion patterns, combining directional movement ratios and conditional logic rules. Specifically, our goal is to be able to identify lane, arch, and turning movements, which are simple but crucial movement patterns. To explain our identification approach mathematically, we use the set-builder notation and logical operators such as \wedge (logical AND), \rightarrow (implies), and $=$ (equal to). Furthermore, we denote the cardinality of a set (number of elements in a set) using the operator $|\cdot|$, that is, the cardinality of the set A would be denoted as $|A|$.

We start by defining directional movements \mathcal{S} as right, left, up, and down motion ($\mathcal{S} \triangleq \{r, l, u, d\}$) according to the drone's perspective. A certain motion vector $v \triangleq (v_x, v_y)$ would be classified in these directions depending on $\theta \triangleq \arctan(v_y/v_x)$, which means that in a given frame, we would classify a motion vector v as pointing right if $\{0 < \theta \leq \pi/4 \wedge 7\pi/4 < \theta \leq 2\pi\}$, left if $\{3\pi/4 < \theta \leq 5\pi/4\}$, up if $\{\pi/4 < \theta \leq 3\pi/4\}$, and finally down if $\{5\pi/4 < \theta \leq 7\pi/4\}$. Each contour c has multiple motion vectors pointing in either of these directions, and identification information lies in the relative proportions of these directions. For each contour c , the proportions of each motion are given by the set $\mathcal{P}^c \triangleq \{p_x^c : \sum p_x^c = 1 \wedge x \in \mathcal{S}\}$, with each motion vector in a contour c counting as one vote towards any direction in \mathcal{S} .

Finally, we are ready to identify dominant motion patterns: Lanes are defined by the straight movement of pedestrians in a specific direction. Thus, a contour c can be classified as a lane under the conditions

$$\begin{aligned} & |\{p_x^c \in \mathcal{P}^c : p_x^c \geq 0.5\}| = 1 \\ & \wedge |\{p_x^c \in \mathcal{P}^c : p_x^c \leq \tau_l\}| = 3 \rightarrow c = \text{lane}, \end{aligned} \quad (1)$$

denoting that if only one direction represents a considerable majority ($\geq 50\%$) of the motion in a contour c , and all other directions individually are less than a certain lane-threshold τ_l , the contour can be classified as a lane. All other motions must

be below the threshold τ_l as, if there exist other significant motions ($\geq \tau_l$) in a given frame, they could signify anomalous or another type of crowd motion. In this case, we found $\tau_l = 0.25$ to provide the best empirical results.

The arch case handles motion patterns where pedestrians have an arching motion in any direction. The condition for arching motion is defined as,

$$|\{p_x^c \in \mathcal{P}^c : p_x^c \leq \tau_a\}| = 1 \rightarrow c = \text{arch}, \quad (2)$$

denoting that an arch consists of three basic movements where one movement connects two opposite movements (such as down, followed by the right, and then followed by up, as in the case of the UCF marathon 3 sequence [18]). Using this pattern, we identify an arch by observing that one of the basic movements should represent less than or equal to some arch threshold, with the other three being dominant motions. For our approach and evaluation, $\tau_a = 0.10$ was experimentally determined to be suitable and used.

Finally, to identify turns, we propose the conditional,

$$|\{p_x^c \in \mathcal{P}^c : p_x^c \geq \tau_t\}| = 2 \rightarrow c = \text{turn}, \quad (3)$$

where a turn is identified if there are two dominant motions in a contour c , denoted by their relative proportion being greater than a turn-threshold τ_t . Furthermore, in this case, there are additional directional conditions to specify, as a turn only occurs when the detected movements are perpendicular to each other. Thus, a turn is classified if the two movements above the threshold are any combination except up and down or right and left, as this would identify a lane rather than a turn. Finally, in this case, $\tau_t = 0.15$ provided the best empirical results.

Beyond identifying motion patterns, this methodology can be easily scaled and expanded. For every contour c , the dominant pattern is calculated at a specific instant, while simultaneously omitting smaller and less significant contours—those less than $1/4$ of the size of the largest contour. Therefore, there could be multiple dominant motions in different areas of the frame if the drone is far enough, and simple heuristics (e.g., number of dominant motions) can be used for inferring and adjusting the drone’s position relative to the crowd.

IV. PERFORMANCE EVALUATION

In this section, we evaluate our approach through both qualitative and quantitative means. We first present the datasets used in our evaluation and then move on to the results.

A. Datasets, Metrics & Experimental Setup

We use multiple datasets and metrics to evaluate our approach. Details are provided below.

Datasets & Metrics: To meticulously evaluate our proposed framework, we incorporate dense crowd datasets, namely the UCF Marathon dataset [18], the TUB CrowdFlow dataset [21], and the Crowd Segmentation and Saliency Detection dataset [20], which encapsulate images of dense crowds with dominant motion patterns, effectively synergizing with our use-case of UAVs and aerial cameras. In particular, TUB

CrowdFlow mimics UAV-recorded videos through dynamic image sequences. Moving toward quantification, our evaluation metric suite encompasses Average Angular Error (AAE), F1-score, and execution times.

Refractory Period: We introduce a novel metric called the ‘refractory period’ that quantifies the time necessary for the drone to stabilize and accurately perceive crowd motion patterns. This helps to better understand the capabilities of the proposed approach, as although it does exhibit robustness against minor jitters (common in UAV videos), it cannot discern motions during substantive drone movement. Therefore, ‘refractory period’ not only evaluates the algorithm’s agility in detecting motion but also sets limits to its expected performance in real-world, during dynamic aerial crowd surveillance.

Experimental Setup: The approach was developed and tested on a Dell Precision 7920 workstation with Python 3.11.0, CUDA 11.8, and FFMPEG 4.2.2 libraries. Comparisons for run-times were evaluated on NVIDIA TX2 and Nano GPUs. Both of them are lightweight (116g and 138g, respectively) embedded GPUs that can be loaded onto small drones to increase computational processing, even though our approach can run on-board the drone without any additional hardware.

B. Qualitative Evaluation

We present a qualitative comparison of our results with those of other dominant motion pattern recognition and identification methods in Fig. 3. In this vein, we identify the main works for the identification of macroscopic crowd patterns, such as Matkovic et al. [8], and Almeida and Jung [11]. However, all of these works use optical flow as a baseline input and then build on top of it using trajectory tracking or other approaches to find the dominant motion pattern. In essence, the quality of the prediction is dependent on the quality of the flow input. The flow methods that these approaches use are Brox flow [9] and Farneback [12]. However, we also compared with Lucas-Kanade [19], and finally the Deep Matching [14] approach as a representative of a data-driven algorithm.

As demonstrated in Fig. 3, the motion estimate (pseudo-flow) derived from H.264, post-implementation of our noise mitigation steps, compares favorably with other recent methods, effectively identifying the dominant motion in a frame. In particular, the resultant shape and structure are of satisfactory quality (especially see performance on Mecca dataset [20]). Furthermore, it is pertinent to acknowledge the limitations introduced by our macroblock-based approach, as there exists an inherent potential for increased error, which can be seen as isolated blocks of incorrect motion interspersed within the dominant pattern. However, the strategic benefits of our approach continue to underscore its pragmatic utility and efficiency in real-world applications.

C. Quantitative Evaluation

We present a detailed quantitative evaluation of our approach, starting with an understanding of its trade-offs. Then we move on to evaluating the identification performance compared with other approaches, and finally end with the execution times.

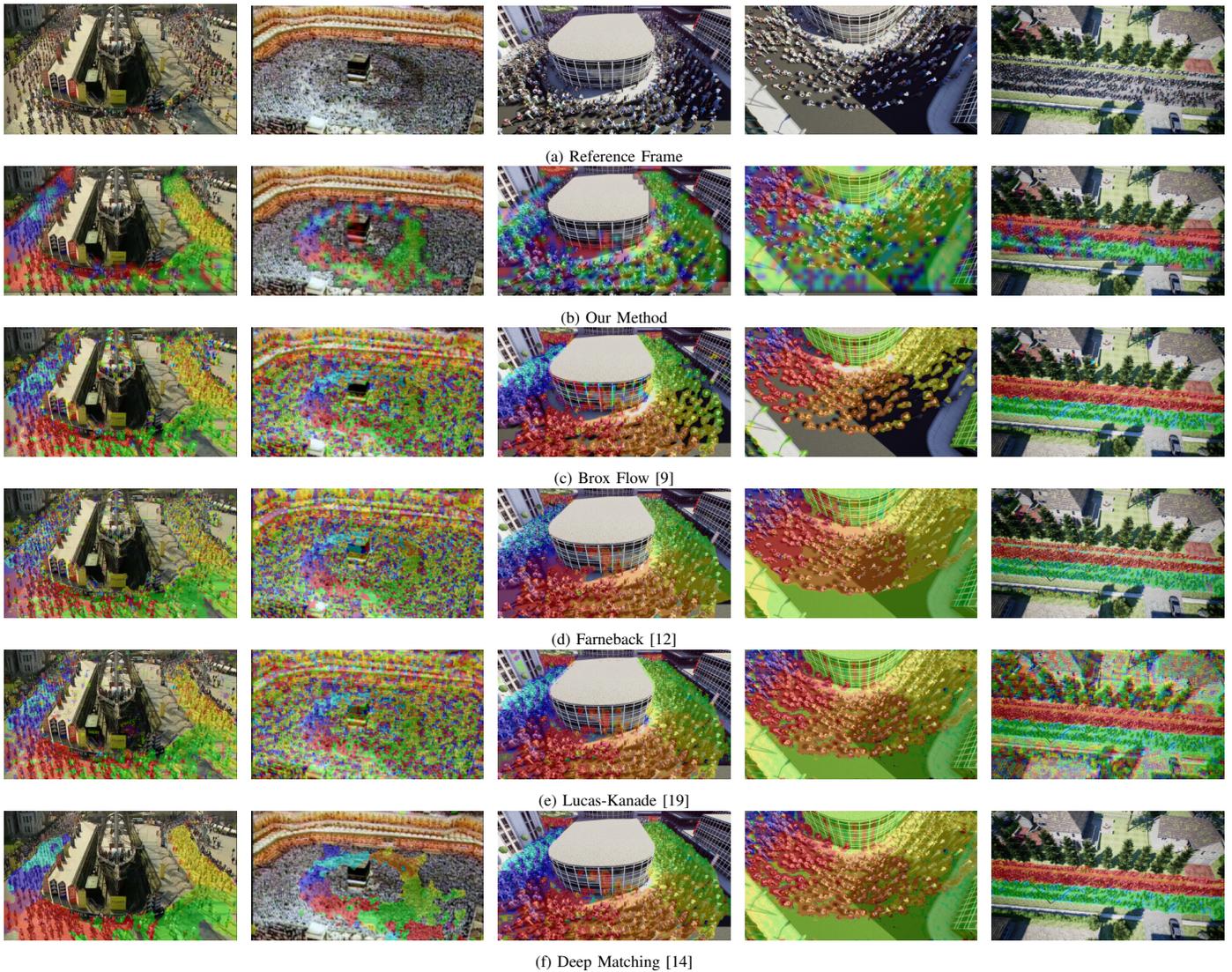


Fig. 3: A qualitative comparison between our approach and other recent approaches to recognize motion patterns in standard datasets. The detected motion is highlighted using colors, while the choice of color represents the direction of the motion. (a) the reference image; (b) our proposed method; (c) the Farneback flow method used by Matkovic et al. [8]; (d) Brox flow used by Almeida and Jung [11]; (e) the Lucas-Kanade flow [19]; and (f) deep-learning approach. The first column is taken from the UCF Marathon dataset: marathon 3 sequence [18], the second column represents the Mecca sequence from the Crowd Saliency dataset [20], the third column represents sequence 5 from the TUB CrowdFlow dataset [21], the fourth column shows the same sequence but simulated as a dynamic (moving) scene as viewed from a UAV, and finally the fifth column represents sequence 3 from the TUB CrowdFlow dataset [21].

Effect of Temporal Averaging: As the proposed approach is centered on real-time lightweight identification of macroscopic patterns, we used the F1 score to evaluate the estimated patterns of our algorithm, and how temporal averaging (used to mitigate noise) affects our predictions. Fig. 4 shows the Gaussian-smoothed raw F1 score across a sample of the frames from the Marathon Dataset for different classified motions. To simulate a turn, we used the marathon 3 sequence and input only the right half of the video sequence, thus representing ‘half marathon 3’ in the legend. Fig. 4 shows that when there is a greater bias toward previous angular values (α), there is more stability in identification, and vice versa.

Transitions in Crowd Pattern: Since our approach is geared

towards mobile crowds, we also performed experiments that involve transitions between crowd patterns by stitching two data sets together and measuring the time it takes for the algorithm to adjust its prediction. As few UAV-centric datasets are available, this was the closest testing method. Fig. 5 shows results from Marathon datasets stitched together. We measure our refractory period using the number of frames until an accuracy above 0.9 is reached (< 30 frames at 30 FPS in Fig. 5). Comparing Figs. 4 and 5, one can appreciate the trade-off involved in choosing α . A higher value of α increases the refractory period but results in a persistently accurate output, while a lower value results in a lower refractory period at the cost of unstable performance. Therefore, α can be managed on

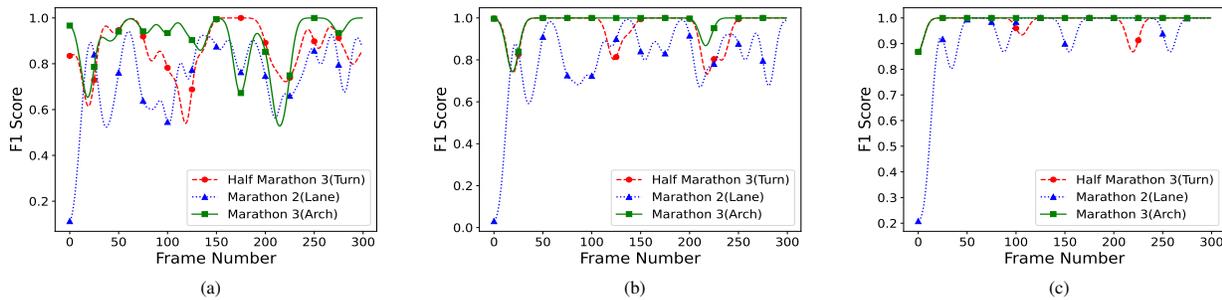


Fig. 4: Gaussian Smoothed F1 Scores with $\sigma = 6$. (a) is for $\alpha = 0.25$, (b) represents $\alpha = 0.50$, and (c) represents $\alpha = 0.75$. An important observation is that giving more weight to new angle values while using older values to slightly correct seems to produce the most consistently accurate results.

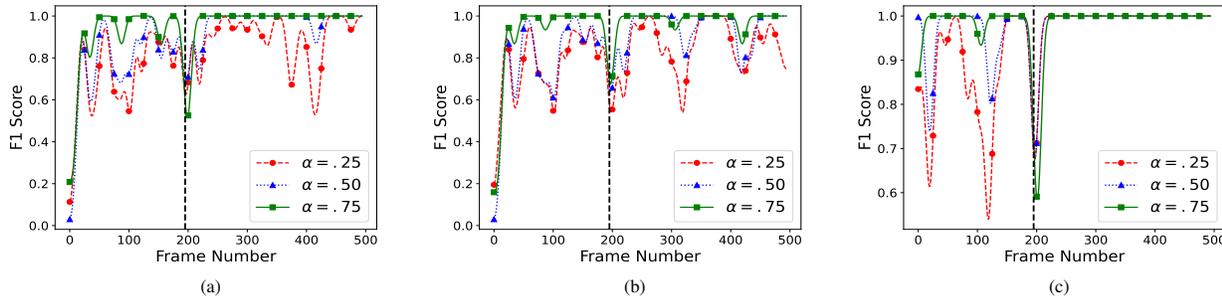


Fig. 5: Smoothed F1 Score for analyzing the refractory period of our proposed method where α represents the ratio of reliance on old information. The dashed line marks the point where the transition from one motion pattern to another occurs. F1 scores are Gaussian-smoothed with $\sigma = 6$. (a) represents the transition from a lane pattern (Marathon 2) to an arch pattern (Marathon 3), (b) from a lane pattern (Marathon 2) to a turning pattern (Half Marathon 3), and (c) from a turning pattern (Half Marathon 3) to an arch pattern (Marathon 3). Giving more weight to older angle values gives the most stability with a slightly greater number of frames to reach the said stability, whereas giving more weight to new angle values seems to show the opposite case.

TABLE I: Average Angular Errors (AAE) and their respective Standard Deviation (STD) on TUB CrowdFlow dataset sequences.

		Our Method		Brox Flow [9]		Farneback [12]		Lucas-Kanade [19]		Deep Matching [14]	
		AAE	STD	AAE	STD	AAE	STD	AAE	STD	AAE	STD
Seq. 1	Static	29.02°	58.66°	19.53°	52.39°	19.66°	52.89°	19.81°	53.59°	20.19°	53.71°
	Dynamic	146.9°	115.7°	235.3°	106.8°	104.3°	145.0°	229.8°	110.8°	230.7°	110.0°
Seq. 2	Static	21.93°	51.10°	18.47°	52.61°	14.73°	49.39°	17.20°	52.28°	14.98°	47.20°
	Dynamic	138.2°	107.3°	170.2°	130.1°	85.06°	133.8°	164.4°	132.8°	164.6°	132.7°
Seq. 3	Static	14.99°	66.76°	38.69°	97.11°	24.95°	86.01°	83.08°	123.8°	22.81°	82.67°
	Dynamic	105.1°	59.27°	100.2°	62.09°	53.35°	62.42°	97.76°	59.80°	96.70°	59.23°
Seq. 4	Static	21.47°	60.06°	31.08°	68.08°	13.77°	51.59°	22.71°	66.32°	25.87°	68.88°
	Dynamic	97.43°	84.45°	225.6°	160.4°	44.49°	114.3°	122.9°	162.5°	206.3°	158.1°
Seq. 5	Static	54.19°	110.2°	69.12°	129.5°	55.53°	123.9°	42.77°	132.0°	66.08°	127.5°
	Dynamic	143.7°	126.2°	192.2°	136.4°	81.23°	143.6°	60.55°	146.3°	188.0°	133.7°

the fly based on application needs, with a lower α favoring rapid decision making with low confidence, and vice versa.

Quantitative Performance Comparison: Table I presents a comparative study of our methodology against state-of-the-art flow estimation techniques in terms of Average Angular Error (AAE) in the TUB CrowdFlow dataset [21]. Consistent with the qualitative results depicted in Fig. 3, our algorithm manifests a commendable performance, notably given its simplicity. Although Farneback stands out as the leading algorithm in this domain, our approach maintains proximity in the error range, even outperforming it once, all along with the advantage of substantially reduced complexity.

Execution Times: Table II illustrates a comparative analysis of the FPS achieved on embedded GPUs, namely NVIDIA TX2

and Nano. Our algorithm records FPS between 30 and 40 for a video stream with resolution 720×400 . Lucas-Kanade appears to outperform our approach in terms of achieved FPS, but the important thing to note here is that—since we leverage the H.264 encoding process— $\sim 95\%$ of the reported time involves processes already running on the drone, while the overhead of our approach is only 2 and 9 ms per frame on TX2 and Nano, respectively. Compared to Matkovic et al. [8], who report 91 ms per frame on a faster AMD Ryzen 7 processor, a runtime of 2 ms still results in a conservative estimate of $45\times$ improvement. Furthermore, approaches like Lucas-Kanade, although similarly fast, lack sharing and will impose a new computational load on the computing board.

TABLE II: Frames Per Second (FPS) on embedded modules NVIDIA TX2 and NVIDIA Nano, respectively. A large FPS (30–40) shows the feasibility of our approach for real-time identification of macroscopic crowd patterns, since that is usually the FPS used in recording video. Although it appears that Lucas-Kanade has a larger FPS in a few instances, it is important to note that $\sim 95\%$ of our reported time involves waiting for shared H.264 processes already running on the drone, with our proposed approach’s overhead only being around 2 and 9 milliseconds for TX2 and Nano, respectively.

		Our Method		Brox Flow [9]		Farneback [12]		Lucas-Kanade [19]		Deep Matching [14]	
		TX2	Nano	TX2	Nano	TX2	Nano	TX2	Nano	TX2	Nano
Seq. 1	Static	41	32	0.3	0.8	12	6	36	23	0.3	0.3
	Dynamic	30	25	0.3	0.8	12	6	37	24	0.3	0.2
Seq. 2	Static	41	32	0.3	0.7	12	6	44	30	0.3	0.3
	Dynamic	31	24	0.3	0.7	12	6	42	29	0.4	0.3
Seq. 3	Static	42	33	0.3	0.8	12	6	44	29	0.3	0.3
	Dynamic	31	25	0.2	0.8	12	6	44	30	0.4	0.3
Seq. 4	Static	38	31	0.3	0.7	12	6	35	24	0.3	0.2
	Dynamic	30	24	0.3	0.8	12	6	41	25	0.4	0.3
Seq. 5	Static	30	25	0.3	0.7	12	6	37	21	0.3	0.3
	Dynamic	28	23	0.3	0.8	12	6	33	30	0.4	0.3

V. CONCLUSION AND FUTURE WORK

We present a novel lightweight approach that makes conscious use of the onboard resources available in a UAV for the detection and identification of dominant crowd motion patterns in real time, taking only a few milliseconds to run on small embedded modules. We also present a thorough analysis and achieve great performance at a negligible cost. However, our algorithm—since it relies on video motion estimation—cannot differentiate between crowd motion and any motion, in general. The presented results have been evaluated for videos with predominant crowd motion, which is what the algorithm is designed for. Experimenting with other kinds of motion might lead to misleading results and is a limitation of the presented work. However, further techniques could be developed to ensure that the input video predominantly presents moving crowds. Still, we believe that this approach is a significant step in the direction of drone-centric intelligence for crowd surveillance, which can be built effortlessly. We will extend this work by solidifying our approach against rapid substantive movements and performing real-life experiments.

Acknowledgements: This work was supported by the NSF RTML Award No. 1937403. We also acknowledge former Rutgers UG students, E. Jagnandan and P. Stecklein, for their contributions to discussions during the capstone design course.

REFERENCES

- [1] E. Vattapparamban, I. Güvenç, A. I. Yurekli, K. Akkaya, and S. Uluagaç, “Drones for smart cities: Issues in cybersecurity, privacy, and public safety,” in *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 216–221, 2016.
- [2] W. Chen, J. Liu, H. Guo, and N. Kato, “Toward robust and intelligent drone swarm: Challenges and future directions,” *IEEE Network*, vol. 34, no. 4, pp. 278–283, 2020.
- [3] B. Piccoli and A. Tosin, “Time-Evolving Measures and Macroscopic Modeling of Pedestrian Flow,” *Archive for Rational Mechanics and Analysis*, vol. 199, pp. 707–738, Mar. 2011.
- [4] C. Feng, P. Han, X. Zhang, B. Yang, Y. Liu, and L. Guo, “Computation offloading in mobile edge computing networks: A survey,” *Journal of Network and Computer Applications*, vol. 202, p. 103366, June 2022.
- [5] ISO Central Secretary, “ISO/IEC 14496-10:2022,” standard, International Organization for Standardization, 2022.
- [6] R. S. De Moraes and E. P. De Freitas, “Multi-uav based crowd monitoring system,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 2, pp. 1332–1345, 2019.
- [7] G. R. Rodríguez-Canosa, S. Thomas, J. Del Cerro, A. Barrientos, and B. MacDonald, “A Real-Time Method to Detect and Track Moving Objects (DATMO) from Unmanned Aerial Vehicles (UAVs) Using a Single Camera,” *Remote Sensing*, vol. 4, pp. 1090–1111, Apr. 2012. Number: 4 Publisher: Molecular Diversity Preservation International.
- [8] F. Matkovic, M. Ivacic-Kos, and S. Ribaric, “A new approach to dominant motion pattern recognition at the macroscopic crowd level,” *Engineering Applications of Artificial Intelligence*, vol. 116, p. 105387, Nov. 2022.
- [9] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, “High accuracy optical flow estimation based on a theory for warping,” in *Computer Vision—ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11–14, 2004. Proceedings, Part IV 8*, pp. 25–36, Springer, 2004.
- [10] L. Zhang, Z. He, M. Gu, and H. Yu, “Crowd segmentation method based on trajectory tracking and prior knowledge learning,” *Arabian Journal for Science and Engineering*, vol. 43, pp. 7143–7152, Dec. 2017.
- [11] I. Almeida and C. Jung, “Crowd flow estimation from calibrated cameras,” *Machine Vision and Applications*, vol. 32, Oct. 2020.
- [12] G. Farneback, “Two-frame motion estimation based on polynomial expansion,” in *Image Analysis: 13th Scandinavian Conference, SCIA 2003 Halmstad, Sweden, June 29–July 2, 2003 Proceedings 13*, pp. 363–370, Springer, 2003.
- [13] L. Kong, C. Shen, and J. Yang, “FastFlowNet: A Lightweight Network for Fast Optical Flow Estimation,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10310–10316, May 2021. ISSN: 2577-087X.
- [14] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, “Deepmatching: Hierarchical deformable dense matching,” *International Journal of Computer Vision*, vol. 120, pp. 300–323, 2016.
- [15] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. v. d. Smagt, D. Cremers, and T. Brox, “FlowNet: Learning Optical Flow with Convolutional Networks,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2758–2766, Dec. 2015. ISSN: 2380-7504.
- [16] R. Mehran, A. Oyama, and M. Shah, “Abnormal crowd behavior detection using social force model,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 935–942, June 2009. ISSN: 1063-6919.
- [17] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, “Overview of the versatile video coding (vvc) standard and its applications,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3736–3764, 2021.
- [18] S. Ali and M. Shah, “Floor fields for tracking in high density crowd scenes,” in *Computer Vision – ECCV 2008* (D. Forsyth, P. Torr, and A. Zisserman, eds.), (Berlin, Heidelberg), pp. 1–14, Springer Berlin Heidelberg, 2008.
- [19] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *IJCAI’81: 7th international joint conference on Artificial intelligence*, vol. 2, pp. 674–679, 1981.
- [20] M. K. Lim, V. J. Kok, C. C. Loy, and C. S. Chan, “Crowd saliency detection via global similarity structure,” in *2014 22nd International Conference on Pattern Recognition*, pp. 3957–3962, 2014.
- [21] G. Schröder, T. Senst, E. Bochinski, and T. Sikora, “Optical flow dataset and benchmark for visual crowd analysis,” 2018.