ADAPTIVE DYNAMIC ACTIVITY MAPPING: A NOVEL APPROACH TO REAL-TIME HEATMAP WITH YOLO v8

Nursiyanto[1], Dona Yuliawati[2], Anggi Andriyadi*[3]
[1,2,3]Department of Information Systems, Faculty of Computer Science
Institut Informatika dan Bisnis Darmajaya, Jl. ZA. Pagar Alam No.93 3514, Bandar Lampung
[1]ikinursiyanto@darmajaya.ac.id, [2]donayuliawati@darmajaya.ac.id, [3]anggi.andriyadi@darmajaya.ac.id

Abstract – Adaptive Dynamic Activity Mapping presents a novel approach for real-time heatmap visualization in object detection systems. Traditional heatmap methods often suffer from ghost effects and fixed kernel sizes, limiting their effectiveness in dynamic scenes. This paper introduces an adaptive approach that integrates with YOLOv8 object detection to provide more accurate and responsive visualization. Performance evaluation across different scenarios demonstrates significant improvements, achieving 97% faster processing in simple scenes while maintaining efficient memory utilization with a 4% reduction compared to traditional methods. While traditional approaches show higher temporal consistency (0.99 vs 0.89), our method eliminates ghost effects and provides better heat uniformity in crowded scenes (0.2495 vs 0.1849). The system employs dynamic kernel generation that adapts to object dimensions, addressing a fundamental limitation of fixed-size kernels in traditional implementations. Experimental results validate the effectiveness of our approach in balancing processing efficiency, visualization quality, and resource utilization, particularly in scenarios requiring accurate temporal representation and clean visualization of object activities.

Keywords - Adaptive heatmap, Object detection, YOLOv8, Real-time visualization, Activity mapping.

I. INTRODUCTION

Object detection is one of computer vision's most critical and essential tasks, aimed at searching for and recognizing objects in images [1]. While object detection provides the foundational analysis, visualization transforms this data into interpretable information that can hide complex details, highlight areas of interest, and provide varying degrees of abstraction [2] [3]. This symbiotic relationship between object detection and visualization [4],[5]. enhances human perception of detection results, facilitates understanding of model behavior, and enables effective system debugging [6] [7]. Consequently, modern visualization technique research has focused on detecting images [8]. Many methods have been published to reveal deep learning mechanisms, but few provide concrete visual cues or expose internal decisions [9]. Therefore, there is a growing need for advanced visualization techniques to provide real-time, adaptive, and efficient representation of object detection results. Among various object detection algorithms, YOLOv8 (You Only Look Once v8) has emerged as a leading framework for real-time applications due to its efficiency and accuracy in handling diverse detection tasks [10]. By integrating YOLOv8 with heatmap visualization, developers can achieve more explainable and interactive object detection systems[11][12], It is because, heatmaps are particularly effective for representing spatial data in object detection architectures, providing users with improved result interpretation capabilities. However, current heatmap visualization techniques face several critical challenges that limit their effectiveness [13]. The primary challenge lies in fixed kernel limitations that prevent adaptation to varying object sizes, coupled with substantial processing overhead in real-time systems. These issues are further compounded by high memory consumption that significantly impacts overall performance [14]. Additionally, developers face considerable difficulty in balancing visualization quality with real-time processing requirements [15] These multifaceted challenges underscore the

pressing need for advanced solutions that can meet critical industry requirements, including real-time processing capability, adaptive visualization mechanisms, and resource-efficient implementation [16]. These requirements are essential for developing practical and effective visualization systems that can operate in real-world conditions.

1. Heatmap Visualization

Heatmap displays can be used to interpret object detection since colour encoding best represents spatial data [17]. They simplify complex information and allow for easy observation of patterns [18]. Visualizations are essential in machine learning and intense learning applications [19]. Visualization and deep learning together provide a comprehensive data analysis [20]. Object detection can classify images and detect the object bounds [21]. Heatmaps provide visual clues for spatial positioning and are commonly used in activity recognition and human pose estimation [22].

2. Limitation of Traditional Heatmap

The traditional heatmap techniques used for object detection have limitations in handling scale and size variations [23]. The heatmaps cannot be scaled with small details and are unsuitable for actual video feeds [24]. It also cannot map temporal information or operate on human behavior datasets [25]. The heatmap generation process requires intensive computational resources, while traditional background models fail to perform effectively in dynamic environments [26]. Traditional previously used methods are not location or climate-specific and present difficulties in real-time object identification [27]. In addition, in the case of object scales, the traditional heatmap visualization for object detection poses a constraint in scaling and is thus not accurate [28]. Static decay rates can cause ghosting or objects vanishing at a very high speed, thus complicating object tracking [29]. High memory usage and computation needs are the main challenges to real-time utilization. It may also be noted that the inability to change stance decreases detection efficiency when the conditions constantly change, leading to missed detections [29]. Some of the problems with the visualization quality, for example, blurring, complicate the interpretation of the heatmap [30]. Enhancement is required to create more accurate, up-to-date, dynamic spatial data models.

3. Propose Method

Traditional heatmap techniques face significant limitations in handling scale and size variations. To address these challenges, we introduce Adaptive Dynamic Activity Mapping (ADAM), which implements adaptive kernel generation to handle varying object sizes while maintaining efficient performance. First, ADAM uses adaptive kernel generation, which adapts to object sizes and replaces static kernel methods. Secondly, ADAM combines with YOLO88 to conduct accurate object detection, especially in the case of different object scales. Third, dynamic decay rates that adapt to area activity levels are used to implement temporal management in ADAM. It removes the typical 'ghost effect' of traditional systems while keeping essential temporal information in less active regions. ADAM optimises for accurate real-time performance with high-quality visualization by minimising memory usage and maximising processing efficiency.

II. SIGNIFICANCE OF STUDY

This section presents the methodology of ADAM, our proposed solution for real-time heatmap visualization in object detection.

1. ADAM Flowchart

As shown in Figure 1, YOLOv8 is used as the object detection engine at the input stage to produce the bounding boxes and confidence scores, which are then used by ADAM's core components to produce heatmaps.

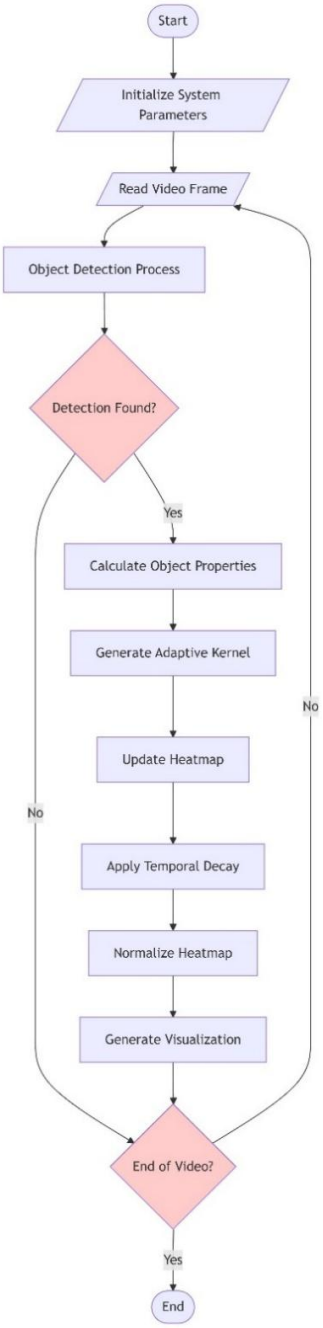

Figure 1. ADAM's Flowchart

The interaction between these components forms a streamlined pipeline where each module performs specialized tasks, starting from object detection through YOLOv8 to the final heatmap

visualization. The following sections detail each component's role and their mathematical foundations, beginning with the YOLOv8 Detection Component.

2. YOLOv8 Detection Company

YOLOv8 is the primary input module in ADAM, providing robust object detection capabilities with three essential parameters: intersection over union (IoU) threshold for Non-Maximum Suppression (NMS), confidence threshold, and model weights. NMS eliminates redundant detections by keeping the highest confidence detection when overlapping areas. BaseDetector standardizes the detection interface and ensures each video frame is processed consistent mathematical framework of the detection process is defined as:

$$D(f) = \{(b_i, c_i, I_i) | i = 1, \dots \dots, n\} \quad (1)$$

Where:

- $D(f)$ represents the detection output for frame f
- $b_i = (x_1, y_1, x_2, y_2)$ is bounding box coordinates
- C_i is the confidence score of the dection
- I_i is the class label
- n is the number of detected objects in the frame

Object filtering is applied through two threshold parameters:

$$P(b_i) = \begin{cases} 1, & if\, c_i \geq \tau_c \, and \, IoU < \tau_{iou} \\ 0, & otherwise \end{cases} \quad (2)$$

Where:

- τ_c = 0.25 is the confidence threshold
- τ_{iou} = 0.45 for NMS threshold
- $P(b_i)$ 1 for retained, 0 for discarded detection

These filters help to feed only the high-confidence detections to the Adaptive Heatmap component, improving accuracy and computational complexity. Moreover, the detector predicts the coordinates of boxes at the spatial level, the confidence score, and the class label, which are passed to the Kernel Generator in the following pipeline.

3. Kernel Generation Component

The Kernel Generator is an important module that synthesizes adequate Gaussian kernels in accordance with the sizes of identified objects. Conventional approaches employ kernels of fixed size, while in ADAM, kernel size changes according to the spatial dimensions of the detected objects, providing a more accurate depiction of activity distribution. The adaptive Gaussian kernel is generated using the following equation:

$$K(x, y) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{(x-\mu_x)^2+(y-\mu_y)^2}{2\sigma^2}\right) \quad (3)$$

Where:
- (x, y) represents the pixel coordinates.
- $(\mu_x . \mu_y)$ is the center of the detected object.
- σ is the adaptive standard deviation.

The adaptive standard deviation σ is calculated based on object size:

$$\sigma = max\ (w, h) \times \alpha \quad (4)$$

Where:
- w is the width of the detected object
- h is the height of the detected object
- $\alpha = 0.15$ is the spatial sigma factor

The generated kernel is then normalized to ensure the sum of all kernel values equals one:

$$K_{norm}(x, y) = \frac{k(x,y)}{\sum_{x,y} K(x,y)} \quad (5)$$

Where:
- $K_{norm}(x, y)$ is the normalized kernel
- $K(x, y)$ is the original kernel
- $\sum_{x,y} K(x, y)$ is the sum of all kernel values

The normalized kernel is then passed to the Adaptive Heatmap Component, which is responsible for managing three key aspects: temporal decay for historical data preservation, activity map updates for current detections, and intensity normalization for balanced visualization.

4. Adaptive Heatmap Component

The Adaptive Heatmap Component manages the temporal evolution and intensity distribution of the activity map through three main processes: temporal decay, activity update, and intensity normalization. For each frame, the heatmap and activity map undergo temporal decay to maintain historical context while emphasizing recent activities. The temporal decay process is defined as:

$$H_t = H_{t-1} \times \beta \quad (6)$$

Where:

- H_t is the heatmap at current time t
- H_{t-1} is the heatmap from previous frame
- $\beta = 0.95$ is the temporal decay factor

For each new detection, the heatmap is updated with the confidence-weighted kernel:

$$H_t(x, y) = H_t(x, y) + K_{norm}(x, y) \times C_i \quad (7)$$

Where $K_{norm}(x, y)$ is the normalized kernel c_i is the detection confidence score Finally, the heatmap undergoes adaptive intensity normalization:

$$H_{norm}(x, y) = clip\left(\frac{h_t(x,y)}{max(h_t)}, \tau_{min}, \tau_{max}\right) \quad (8)$$

Where:
- τ_{min} 0.2 is the minimum intensity threshold
- τ_{max} 1.0 is maximum intensity threshold
- $clip(x, a, b)$ clamps the value x between a and b

This normalized heatmap provides a balanced visualization where the intensity reflects both current detections and historical activities while maintaining a clear distinction between active and inactive regions.

5. Testing Evaluation

The evaluation of ADAM's algorithm encompassed a comprehensive comparative analysis against traditional heatmap visualization methods when integrated with YOLOv8, examining its performance across two distinct experimental scenarios: detection with limited objects and detection with multiple objects. This comparative approach was strategically designed to assess ADAM's adaptive capabilities and limitations under varying object density conditions, utilizing both a custom dataset comprising 5,000 labelled images for smoking activity detection with limited objects (specifically two people) from top-down views [31]. and the COCO dataset with YOLOv8n for multiple object detection scenarios [32]. This methodologically rigorous dual-dataset approach facilitated a thorough evaluation of the system's performance across different object densities, enabling the identification of optimal operating conditions and a systematic understanding of system limitations, thereby providing a comprehensive assessment of ADAM's practical applicability in diverse detection scenarios. Furthermore, Testing was performed on an NVIDIA RTX 3060 12GB, chosen for its high-performance capabilities to evaluate ADAM's real-time processing efficiency and scalability under diverse scenarios.

6. Performance Metric

To comprehensively evaluate ADAM's adaptive heatmap performance compared to traditional heatmap methods, we were used several key metrics were used, in the table 1. These metrics collectively provide a robust framework for analyzing ADAM's performance, highlighting its adaptability and visualization quality.

TABLE 1.
PERFORMANCE METRIC VARIABLE

Metric	Description
Processing Time	Measures the average time per frame to assess real-time performance.
Temporal Consistency	Evaluates the stability of heatmap transitions across consecutive frames.
Memory Usage	Analyzes average and peak memory consumption during heatmap generation.
Spatial Adaptation	Tests the system's ability to dynamically adjust to varying object sizes.
Heatmap Uniformity	Evaluates the evenness of intensity distribution in the heatmap.
Gradient Smoothness	Measures the clarity of intensity transitions within the heatmap.

III. RESULTS AND DISCUSSION

This section presents the experimental results of ADAM and analyzes its performance compared to traditional heatmap methods. The evaluation focuses on processing efficiency, visualization quality, and the effectiveness of YOLOv8 integration. Through comprehensive testing across different scenarios and datasets, we demonstrate ADAM's capabilities in real-time object detection visualization. Two distinct test scenarios were conducted to evaluate ADAM's performance: a limited-object scenario focusing on specific activity detection and a multiple-object scenario in crowded environments. These contrasting conditions allow us to assess both the algorithm's adaptability and its performance boundaries in real-world applications.

1. Visual Analysis and Detection Results

As shown in figure 2, in crowded scenes, ADAM demonstrates both its strengths and limitations. A significant performance impact is observed as the frame rate drops from 25 FPS to 4 FPS, representing an 84% decrease in processing speed. This substantial reduction is primarily attributed to the increased computational load when YOLOv8 processes multiple objects while simultaneously generating adaptive heatmaps for each detection.

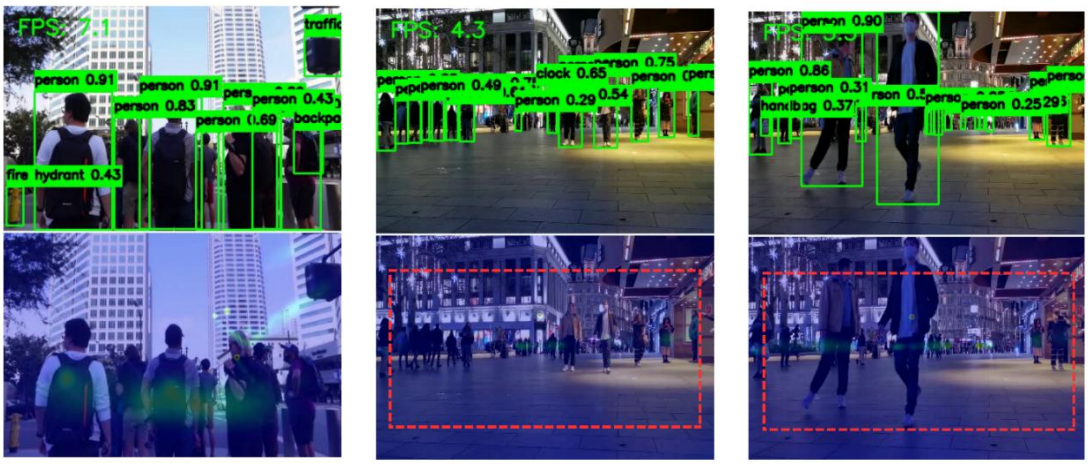

Figure 2. Sequential visualization of ADAM's ghost effect elimination: (a) initial adaptive heatmap generation based on current detections, (b) transition phase showing active clearing of previous detections, and (c) generation of new heatmap without residual traces, demonstrating efficient temporal adaptation in crowded scenes.

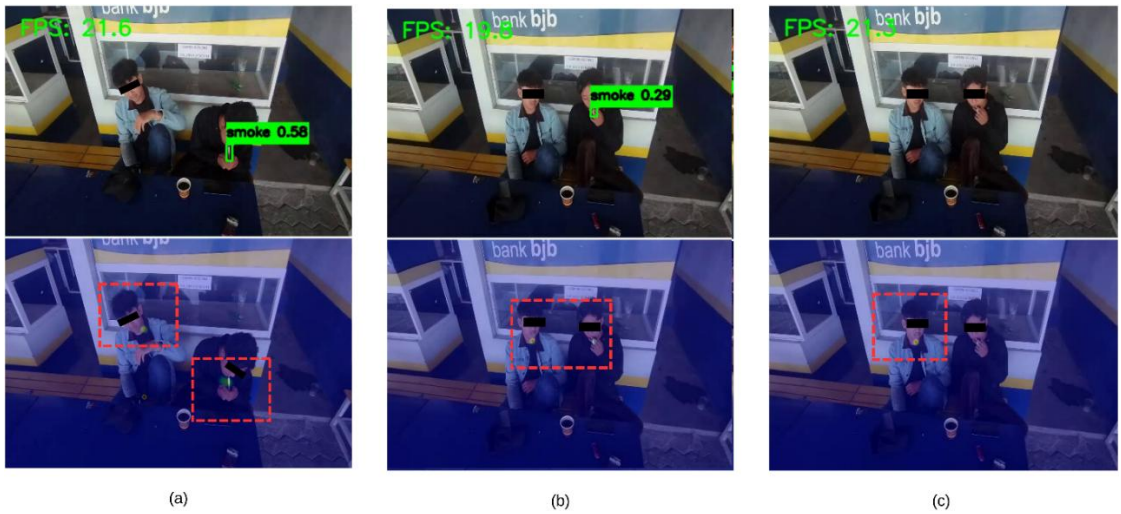

Figure 3. ADAM's performance in limited-object smoking activity detection scenario: (a) initial detection of smoking activity with adaptive heatmap generation in the region of interest (highlighted in red box), (b) transition phase showing

temporal adaptation as the scene changes, and (c) generation of new activity heatmap with consistent performance at 21.6 FPS.

The red boxes indicate areas where objects have left the detection frame, demonstrating how their corresponding heatmap signatures immediately disappear without leaving residual traces. This clean transition between frames highlights ADAM's ability to eliminate ghost effects while maintaining visualization clarity in multi-object scenarios. Furthermore, In the smoking detection case in Figure 2, ADAM demonstrates robust performance in tracking specific activities with limited objects. The sequence shows consistent FPS rates of 21.6, indicating stable real-time performance. The adaptive heatmap effectively focuses on the smoking activity area, with the red boxes highlighting the regions of interest where the activity is detected. The temporal adaptation is particularly noticeable as the heatmap smoothly updates to reflect changes in smoking behavior without leaving any residual heat signatures from previous detections.

2. Performance Analysis

As shown on table 2, our comprehensive evaluation of ADAM's performance across different scenarios reveals interesting patterns in processing efficiency and resource utilization. In multi-object crowded scenarios, the system demonstrates an average processing time of 0.0232 seconds per frame, while in limited-object scenarios smoking detection, it achieves significantly faster processing at 0.0006 seconds per frame. This 97% reduction in processing time for limited-object scenarios illustrates ADAM's efficient scaling based on scene complexity. Furthermore, temporal consistency remains robust across both scenarios, with crowded scenes achieving 0.8990 and limited-object scenes slightly higher at 0.9015. This minimal variation (less than 0.3%) indicates that ADAM maintains reliable temporal adaptation regardless of scene complexity. Memory usage remains constant at 1,843,200 units across both scenarios, suggesting efficient memory management and optimal resource allocation. Notably, spatial adaptation shows contrasting behaviour between scenarios.

TABLE 2.
ADAM PERFORMANCE METRIC

Metric	Multi Object	Limited Object
Processing Time	0.0232	0.0006
Temporal Consistency	0.8990	0.9015
Memory Usage	1,843,200	1,843,200
Spatial Adaptation	-0.4736	0.1412
Heatmap Uniformity	0.2495	0.0000
Gradient Smoothness	0.9826	0.0000
Total Frames	1,109	1,800

However, based on Table 2, In crowded scenes, we observe a negative spatial adaptation improvement of -0.4736, indicating increased computational challenges in managing multiple adaptive kernels. However, in limited-object scenarios, the positive improvement of 0.1412 demonstrates ADAM's optimal performance when tracking specific activities. Heat distribution metrics in crowded scenes show moderate uniformity (0.2495) with excellent gradient smoothness (0.9826), contributing to visually coherent heatmap generation. The absence of these metrics in limited-object scenarios (both at 0.0000) reflects the system's focused adaptation to specific detection tasks.

3. Comparative Analysis with Traditional Heatmap

This section presents a comparative analysis between ADAM and traditional heatmap approaches across different scenarios. To ensure a fair comparison, both methods were evaluated under identical conditions using the same datasets and evaluation metrics.

TABLE 3.
TRADITIONAL HEATMAP PERFORMANCE METRIC

Metric	Multi Object	Limited Object
Processing Time	0.0220	0.0230
Temporal Consistency	0.9993	0.9982
Memory Usage	1,920,000	1,920,000
Spatial Adaptation	N/A	N/A
Heatmap Uniformity	0.1849	-1.2746
Gradient Smoothness	0.9710	0.9968

Based on Table 2 and 3, while traditional methods maintain consistent processing times (~0.022s) regardless of scene complexity, ADAM shows adaptive processing with exceptional performance in limited-object scenes (0.0006s) and comparable performance in crowded scenes (0.0232s). Memory management analysis shows ADAM achieving 4% lower utilization across all scenarios (1,843,200 vs 1,920,000 units). Furthermore, as shown in Figure 4, traditional heatmap methods demonstrate a significant limitation in temporal adaptation. The visualization shows prominent ghost effects where activity heat signatures persist long after subjects have moved, creating large areas of residual heat that do not reflect current scene activities. While this results in higher temporal consistency scores (0.99 vs 0.89), it actually represents a disadvantage in real-world applications. ADAM, in contrast, shows better heat uniformity in crowded scenes (0.2495 vs 0.1849) with cleaner temporal transitions.

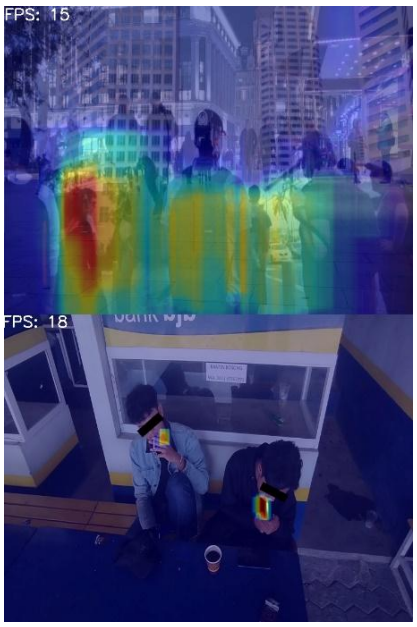

Figure 4. Comparison of heatmap generation between traditional and ADAM approaches in a smoking detection scenario. The top image shows traditional heatmap with prominent ghost effects, where heat signatures persist and spread across large areas even after subjects have moved.

A key distinction lies in spatial adaptation capabilities. Traditional implementations use fixed-size Gaussian kernels, while ADAM employs dynamic kernel generation adapting to object dimensions. This fundamental difference is reflected in the absence of spatial adaptation metrics in traditional heatmaps and visually evident in the more focused, accurate heat distribution of ADAM compared to the diffused patterns in traditional approaches. These findings show ADAM's advantages in processing speed, memory efficiency, and adaptive capabilities, making it suitable for applications

requiring rapid response and clean visualization. Traditional approaches remain preferable where consistent processing speed is prioritized over adaptive optimization.

In addition, the experimental results reveal important insights about adaptive heatmap implementations in real-time object detection systems. The significant performance variation between simple and complex scenes (0.0006s vs 0.0232s) indicates that scene complexity remains a critical factor in real-time processing capabilities. This finding highlights the importance of context-aware optimization in visual analytics systems. While the 4% improvement in memory efficiency might appear modest, it demonstrates that adaptive approaches can maintain lower resource requirements even while providing more sophisticated functionality. This efficiency becomes particularly significant in extended operations or resource-constrained environments, where traditional methods' higher memory usage could limit deployment options.

The trade-off between temporal consistency and ghost effect elimination represents a fundamental challenge in heatmap visualization. Traditional methods' higher temporal consistency (0.99) actually impedes accurate activity representation, while ADAM's lower score (0.89) reflects its success in balancing persistence with adaptability. This finding challenges the conventional assumption that higher temporal consistency necessarily indicates better performance. These findings suggest that adaptive approaches offer tangible advantages in real-world applications, particularly where accurate temporal representation and efficient resource utilization are prioritized over raw processing speed. However, the performance degradation in complex scenes indicates that further optimization may be necessary for high-density scenarios.

IV. CONCLUSIONS

This paper presents ADAM, a novel approach for adaptive heatmap visualization in real-time object detection. Through experimental evaluation, we have demonstrated significant improvements in processing speed for simple scenes and consistent memory efficiency across different scenarios, while successfully eliminating ghost effects that plague traditional methods. Our results show that adaptive approaches can effectively balance performance and visualization quality, achieving a 97% improvement in processing speed for simple scenes while maintaining comparable performance in complex scenarios. The trade-off between temporal consistency and accurate activity representation highlights the importance of context-aware adaptation in visual analytics systems. Future research should focus on three key areas: optimization for complex scenes, integration of machine learning approaches for kernel adaptation, and implementation of parallel processing techniques. First, optimization strategies for complex scenes could enhance performance without sacrificing adaptation quality. Second, the integration of machine learning approaches might enable more intelligent kernel adaptation mechanisms. Finally, investigating parallel processing techniques could help maintain real-time performance in high-density scenarios while preserving the benefits of adaptive visualization.

REFERENCES

[1] A. S. Rao *et al.*, “Real-time monitoring of construction sites: Sensors, methods, and applications,” *Autom Constr*, vol. 136, p. 104099, Apr. 2022, doi: 10.1016/J.AUTCON.2021.104099.

[2] C. Linse, H. Alshazly, and T. Martinetz, “A walk in the black-box: 3D visualization of large neural networks in virtual reality,” *Neural Comput Appl*, vol. 34, no. 23, pp. 21237–21252, Dec. 2022, doi: 10.1007/S00521-022-07608-4/TABLES/8.

[3] K. Bayoudh, R. Knani, F. Hamdaoui, and A. Mtibaa, “A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets,” *The Visual Computer 2021 38:8*, vol. 38, no. 8, pp. 2939–2970, Jun. 2021, doi: 10.1007/S00371-021-02166-7.

[4] X. Wang, L. Zhu, Y. Wu, and Y. Yang, "Symbiotic Attention for Egocentric Action Recognition With Object-Centric Alignment," *IEEE Trans Pattern Anal Mach Intell*, vol. 45, no. 6, pp. 6605–6617, Jun. 2023, doi: 10.1109/TPAMI.2020.3015894.

[5] F. Li, Z. Yang, and Y. Gui, "SES-YOLOv8v5: small object graphics detection and visualization applications," *Visual Computer*, pp. 1–14, Aug. 2024, doi: 10.1007/S00371-024-03591-0/FIGURES/10.

[6] S. Chen, J. Yu, and S. Wang, "One-dimensional convolutional neural network-based active feature extraction for fault detection and diagnosis of industrial processes and its understanding via visualization," *ISA Trans*, vol. 122, pp. 424–443, Mar. 2022, doi: 10.1016/J.ISATRA.2021.04.042.

[7] E. Kim, D. Gopinath, C. S. Păsăreanu, and S. A. Seshia, "A programmatic and semantic approach to explaining and debugging neural network based object detectors," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 11125–11134, 2020, doi: 10.1109/CVPR42600.2020.01114.

[8] J. Chai, H. Zeng, A. Li, and E. W. T. Ngai, "Deep learning in computer vision: A critical review of emerging techniques and application scenarios," *Machine Learning with Applications*, vol. 6, p. 100134, Dec. 2021, doi: 10.1016/J.MLWA.2021.100134.

[9] X. Li *et al.*, "Interpretable deep learning: interpretation, interpretability, trustworthiness, and beyond," *Knowl Inf Syst*, vol. 64, no. 12, pp. 3197–3234, Dec. 2022, doi: 10.1007/S10115-022-01756-8/TABLES/3.

[10] S. G. Verma, S. Maurya, H. Pant, A. K. Yadav, S. Thomas Varghese, and R. Garg, "YOLOv8: Redefining Real-Time Object Detection Accuracy and Efficiency," *2024 IEEE 3rd World Conference on Applied Intelligence and Computing (AIC)*, pp. 1123–1129, Jul. 2024, doi: 10.1109/AIC61668.2024.10731055.

[11] A. Karasmanoglou, M. Antonakakis, and M. Zervakis, "Heatmap-based Explanation of YOLOv5 Object Detection with Layer-wise Relevance Propagation," *IST 2022 - IEEE International Conference on Imaging Systems and Techniques, Proceedings*, 2022, doi: 10.1109/IST55454.2022.9827744.

[12] S. Phatangare, S. Kate, D. Khandelwal, A. Khandetod, and A. Kharade, "Real-time Human Activity Detection using YOLOv7," *7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), I-SMAC 2023 - Proceedings*, pp. 1069–1076, 2023, doi: 10.1109/I-SMAC58438.2023.10290168.

[13] T. Yamauchi and M. Ishikawa, "SPATIAL SENSITIVE GRAD-CAM: VISUAL EXPLANATIONS FOR OBJECT DETECTION BY INCORPORATING SPATIAL SENSITIVITY," *Proceedings - International Conference on Image Processing, ICIP*, pp. 256–260, 2022, doi: 10.1109/ICIP46576.2022.9897350.

[14] R. Wu, X. Xiao, G. Hu, H. Zhao, H. Zhang, and Y. Peng, "DetOH: An Anchor-Free Object Detector with Only Heatmaps," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 14177 LNAI, pp. 152–167, 2023, doi: 10.1007/978-3-031-46664-9_11/TABLES/4.

[15] S. Zhao, L. Zhou, W. Wang, D. Cai, T. L. Lam, and Y. Xu, "Toward Better Accuracy-Efficiency Trade-Offs: Divide and Co-Training," *IEEE Transactions on Image Processing*, vol. 31, pp. 5869–5880, 2022, doi: 10.1109/TIP.2022.3201602.

[16] W. E. Villegas, S. Sanchez-Viteri, and S. Lujan-Mora, "Real-Time Recognition and Tracking in Urban Spaces Through Deep Learning: A Case Study," *IEEE Access*, vol. 12, pp. 95599–95612, 2024, doi: 10.1109/ACCESS.2024.3426295.

[17] X. Hao, J. Tian, H. Ding, K. Zhao, and M. Gen, "An enhanced two phase estimation of distribution algorithm for solving scheduling problem," *https://doi.org/10.1080/17509653.2022.2085205*, pp. 1–8, Jun. 2022, doi: 10.1080/17509653.2022.2085205.

[18] Y. Qi, H. Zhang, and J. Liu, "More accurate heatmap generation method for human pose estimation," *Multimed Syst*, vol. 30, no. 4, pp. 1–11, Aug. 2024, doi: 10.1007/S00530-024-01390-0/TABLES/5.

[19] S. Dubey and M. Dixit, "A comprehensive survey on human pose estimation approaches," *Multimedia Systems 2022 29:1*, vol. 29, no. 1, pp. 167–195, Aug. 2022, doi: 10.1007/S00530-022-00980-0.

[20] P. Chen, Q. Li, S. Biaz, T. Bui, and A. Nguyen, "gScoreCAM: What Objects Is CLIP Looking At?," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and*

Lecture Notes in Bioinformatics), vol. 13844 LNCS, pp. 588–604, 2023, doi: 10.1007/978-3-031-26316-3_35/FIGURES/4.

[21] Z. Xu, E. Hrustic, and D. Vivet, "CenterNet Heatmap Propagation for Real-Time Video Object Detection," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12370 LNCS, pp. 220–234, 2020, doi: 10.1007/978-3-030-58595-2_14/TABLES/5.

[22] Y. Qi, H. Zhang, and J. Liu, "More accurate heatmap generation method for human pose estimation," *Multimed Syst*, vol. 30, no. 4, pp. 1–11, Aug. 2024, doi: 10.1007/S00530-024-01390-0/TABLES/5.

[23] L. Yan, Y. Qin, and J. Chen, "Scale-Balanced Real-Time Object Detection With Varying Input-Image Resolution," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 1, pp. 242–256, Jan. 2023, doi: 10.1109/TCSVT.2022.3198329.

[24] Z. Xu, E. Hrustic, and D. Vivet, "CenterNet Heatmap Propagation for Real-Time Video Object Detection," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12370 LNCS, pp. 220–234, 2020, doi: 10.1007/978-3-030-58595-2_14/TABLES/5.

[25] C. Chen, G. Wang, C. Peng, Y. Fang, D. Zhang, and H. Qin, "Exploring Rich and Efficient Spatial Temporal Interactions for Real-Time Video Salient Object Detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 3995–4007, 2021, doi: 10.1109/TIP.2021.3068644.

[26] H. F. Ates, A. Siddique, and B. Gunturk, "HMRN: regression network to detect and track small objects in wide-area motion imagery," *Signal Image Video Process*, vol. 17, no. 1, pp. 39–45, Feb. 2023, doi: 10.1007/S11760-022-02201-7/TABLES/5.

[27] R. Dong, S. Yin, L. Jiao, J. An, and W. Wu, "ASIPNet: Orientation-Aware Learning Object Detection for Remote Sensing Images," *Remote Sensing 2024, Vol. 16, Page 2992*, vol. 16, no. 16, p. 2992, Aug. 2024, doi: 10.3390/RS16162992.

[28] J. Mu, Q. Su, X. Wang, W. Liang, S. Xu, and K. Wan, "A small object detection architecture with concatenated detection heads and multi-head mixed self-attention mechanism," *J Real-time Image Process*, vol. 21, no. 6, pp. 1–15, Dec. 2024, doi: 10.1007/S11554-024-01562-1/TABLES/14.

[29] H. F. Ates, A. Siddique, and B. Gunturk, "HMRN: regression network to detect and track small objects in wide-area motion imagery," *Signal Image Video Process*, vol. 17, no. 1, pp. 39–45, Feb. 2023, doi: 10.1007/S11760-022-02201-7/TABLES/5.

[30] A. Marchand, P. Balbastre, I. Ripoll, M. Masmano, and A. Crespo, "Memory resource management for real-time systems," *Proceedings - Euromicro Conference on Real-Time Systems*, pp. 201–210, 2007, doi: 10.1109/ECRTS.2007.18.

[31] Anggi Andriyadi, "Smoking Detection - v1 2024-08-17 11:52pm," Roboflow. Accessed: Nov. 22, 2024. [Online]. Available: https://universe.roboflow.com/husky-wrfw6/smoking-detection-3i0zg-kcrv5/dataset/1

[32] T. Y. Lin *et al.*, "Microsoft COCO: Common Objects in Context," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8693 LNCS, no. PART 5, pp. 740–755, May 2014, doi: 10.1007/978-3-319-10602-1_48.

ACKNOLOWGDMENT

This work was sponsored by Institut Informatika dan Bisnis Darmajaya (IIB Darmajaya) under grant No. SK.0536/DMJ/REK/LPPM.X.2024