JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGY
https://e-journal.uum.edu.my/index.php/jict

How to cite this article:
Yuliawati, D., Saleh, S., Irianto, S. Y. & Andriyadi, A. (2026). Motion-pattern-based multi-object tracking for real-time edge computing deployment. *Journal of Information and Communication Technology, 25*(2), 96-121. https://doi.org/10.32890/jict2026.25.2.5

Motion-Pattern-Based Multi-Object Tracking for Real-Time Edge Computing Deployment

[1]Dona Yuliawati, [2]Sushanty Saleh, [3]Suhendro Yusuf Irianto & [4]Anggi Andriyadi
[1,2&4]Department of Information System,
Institut Informatika dan Bisnis Darmajaya, Indonesia
[3]Department of Informatics, Institut Informatika dan Bisnis Darmajaya, Indonesia

[*1]donayuliawati@darmajaya.ac.id
[2]sushantysaleh@darmajaya.ac.id
[3]suhendro@darmajaya.ac.id
[4]anggi.andriyadi@darmajaya.ac.id
*Corresponding author

Received: 28/9/2025 Revised: 22/4/2026 Accepted: 27/4/2026 Published: 30/4/2026

ABSTRACT

Real-time object tracking in embedded systems requires algorithms that balance processing performance with energy efficiency for practical deployment. This study evaluates the performance of the TG-Lite algorithm on Jetson Nano embedded hardware for surveillance applications requiring a minimum processing rate of 10 frames per second (FPS). The methodology involves comprehensive performance analysis, measuring frame-processing rates, energy consumption patterns, and system resource utilisation across video-file and live-stream scenarios. TG-Lite achieves 15.51 FPS processing capability with 11.24 watts (W) power consumption, meeting real-time processing requirements for embedded surveillance deployment. The algorithm demonstrates consistent performance across different input conditions, maintaining stable processing rates on both video files (10.86 FPS) and live streams (15.51 FPS). TG-Lite exhibits efficient resource utilisation, with moderate central processing unit (CPU) usage (17.4% average) and effective graphics processing unit (GPU) acceleration. The energy analysis reveals TG-Lite's 1.38 frames-per-watt efficiency, indicating effective allocation of computational resources for real-time processing demands. Performance evaluation shows that TG-Lite maintains stable tracking continuity, with an average track length of 35.8 frames and a consistent confidence distribution pattern. These results demonstrate the suitability of the proposed approach for embedded computer vision applications under the evaluated edge constraints.

Keywords: Computer vision, embedded systems, energy efficiency, object tracking, real-time processing.

INTRODUCTION

The proliferation of Internet of Things (IoT) devices worldwide has reached unprecedented levels, with the number of connected IoT devices growing 13% to 18.8 billion globally in 2024 and projected to exceed 40.6 billion by 2034 (Satyajit Sinha, 2023). This exponential growth occurs alongside the urgent need for intelligent surveillance systems within the global road safety context, where the World Health Organisation reports that road traffic deaths reach 1.19 million people annually worldwide (World Health Organization, 2023), with road traffic crashes being the leading cause of death for individuals aged 5-29 years (Global Road Safety | Transportation Safety | CDC, n.d.).

In this context, UN-Habitat, through the World Smart Cities Outlook 2024, emphasises the importance of people-centred smart city technologies to create sustainable urban transformation (World Smart Cities Outlook 2024 | UN-Habitat, n.d.). As urban populations continue to expand, projected to increase from 55% currently to 68% by 2050 (Quasim et al., 2021). The demand for intelligent traffic monitoring systems that operate efficiently on resource-constrained edge devices has become increasingly critical (Ansari & Singh, 2022; OECD Urban Studies, 2023). The global edge computing market, valued at USD 432.94 billion in 2024 and projected to reach USD 5,132.29 billion by 2034 (Precedence Research, 2026). This underscores the growing importance of deploying real-time processing capabilities directly at the network edge rather than relying on centralised infrastructure.

While the Ultralytics YOLO family (YOLOv8-v11) has established itself as a leading object detection framework, fundamental architectural limitations arise when deploying detection-only algorithms for tracking. The segmentation of detection and tracking tasks in Tracking by Detection (TBD) methods into different training processes leads to global inconsistency, which degrades overall performance (Guan et al., 2025), highlighting the inherent disconnect between frame-independent detection and the requirements of temporal tracking. This limitation is compounded by the computational overhead associated with YOLO's architectural complexity; YOLOv8 offers five variants, with the smallest comprising 225 layers, and utilising these models for specific object sizes in resource-constrained applications may entail high computational costs (Rasheed & Zarkoosh, 2025) Furthermore, existing detection approaches face persistent challenges, including missed detections due to occlusion, illumination variations, and the dynamic nature of object behaviour, contributing to temporal inconsistencies with high false positives and false negatives (Nasir et al., 2025).

Successive YOLO iterations have introduced notable architectural improvements - YOLOv8 with Path Aggregation Network and Dynamic Kernel Attention, YOLOv9 with multi-level auxiliary feature extraction, YOLOv10 with a lightweight classification head, and YOLOv11 with the C3k2 block and C2PSA for spatial attention (Sapkota et al., 2025; Varghese & Sambath, 2024) - yet none of these versions incorporates inherent mechanisms for maintaining object identity across temporal sequences. The detection process treats each frame as an independent inference task, without consideration for object continuity or identity preservation. This absence of temporal coherence creates systematic counting errors: duplicate counting when objects are detected intermittently across frames, and undercounting when temporary occlusions or lighting variations cause detection failures in consecutive frames (Murat & Kiran, 2025). The computational resources required for frame-by-frame processing without temporal optimisation result in inefficient use of processing capabilities in resource-constrained edge computing environments.

This fundamental gap between detection capabilities and tracking requirements necessitates the integration of lightweight tracking algorithms that bridge temporal inconsistencies while preserving computational efficiency for real-time applications on edge devices. Tracking provides the object identity maintenance missing in pure detection approaches, enabling the transformation of fragmented detection data into consistent tracking information essential for monitoring applications in traffic safety and urban surveillance contexts.

To address these limitations, TG-Lite is positioned as a deployment-oriented component within edge-based ICT systems rather than solely as a computer vision algorithm. By prioritising bounded computational complexity and predictable processing latency, TG-Lite supports real-time information processing on resource-constrained edge nodes. This design aligns with distributed edge intelligence paradigms, where object tracking and preliminary decision-making are performed locally, reducing dependency on centralised processing and facilitating integration into smart surveillance and traffic management infrastructures. TG-Lite integrates three core components: Adaptive Confidence Fusion (ACF) for dynamically weighting detection confidence and motion consistency using sigmoid activation functions; Smart History Management (SHM) for temporal stabilisation through tanh-based smoothing adapted to track maturity; and History Bonus Mechanism (HBM) for reinforcing consistently performing tracks against premature termination. These components operate within a greedy matching strategy with O(n^2) computational complexity, specifically designed for minimal latency and predictable resource usage on platforms such as Jetson Nano.

RELATED WORKS

The Simple Online and Realtime Tracking (SORT) algorithm introduced by Bewley et al. remains one of the most influential frameworks in multi-object tracking (MOT), utilising Kalman filtering for motion prediction and the Hungarian algorithm for data association with $O(n^3)$ computational complexity (Bewley et al., 2016). The algorithm represents each object using a discrete Kalman Filter that estimates state vectors through linear stochastic difference equations, achieving updates at 260 Hz, which is over twenty times faster than other trackers at the time. Building upon this foundation, DeepSORT extends the basic SORT framework by incorporating deep appearance features through convolutional neural networks, integrating appearance information to improve tracking performance through longer periods of occlusions (Wojke et al., 2017). However, despite these improvements, the Mahalanobis distance computation tends to favour tracks that have not been updated in recent frames due to covariance matrix growth for unassigned tracks.

While SORT and DeepSORT have established the foundation for modern MOT systems, subsequent developments have focused on improving their robustness and accuracy. StrongSORT, introduced by Du et al. (2023) significantly enhances DeepSORT across multiple dimensions, including object detection, feature embedding, and trajectory association, thereby establishing a strong baseline for the MOT community. This improvement addresses two inherent problems in MOT systems - missing association and missing detection - while maintaining the fundamental Kalman filter and Hungarian algorithm architecture.

Recent advances in MOT have introduced several motion-centric approaches that further improve upon the SORT-based paradigm. ByteTrack (Zhang et al., 2022) proposes a two-stage association strategy that leverages low-confidence detections to recover missed tracks, achieving state-of-the-art performance on MOT17 by associating almost every detection box rather than discarding low-score

ones. This approach demonstrates that utilising all detection outputs, including those below conventional confidence thresholds, can significantly reduce false negatives. OC-SORT (Cao et al., 2023) addresses the degradation of Kalman filter estimates during occlusion periods by using an observation-centric re-update mechanism that corrects accumulated estimation errors when objects reappear after a temporary disappearance. BoT-SORT (Aharon et al., 2022) extends the tracking pipeline further by incorporating camera motion compensation alongside Kalman prediction and appearance features, improving robustness in scenarios with camera movement. While these methods significantly improve tracking accuracy on standard benchmarks, they share a common architectural reliance on Kalman filter state estimation with matrix operations and on Hungarian algorithm-based assignment with $O(n^3)$ complexity, making their computational requirements challenging for resource-constrained edge platforms.

Comprehensive reviews highlight the ongoing challenges in the field, as MOT research has evolved from traditional statistical model-based methods to incorporate deep learning architectures, yet persistent challenges remain with accuracy due to occlusion, lighting variation, and camera movement (Guan et al., 2025). Recent enhancements to these baseline methods, such as EF-StrongSORT, demonstrate improvements in challenging scenarios but continue to rely on the same computational foundations (Khalil et al., 2025). To address some of these challenges, enhanced tracking approaches have explored incorporating multi-modal information, such as combining 3D point clouds and colour images to improve robustness, though these methods require substantial computational resources (An et al., 2023). However, these advances have generally maintained the core computational architecture of Kalman filtering and Hungarian assignment.

Recognising these deployment challenges, researchers have developed various strategies to reduce computational overhead. Several lightweight approaches have emerged to address resource constraints. MobileNet-JDE achieves model size reductions of up to 70% while maintaining high accuracy on NVIDIA Jetson Orin Nano, demonstrating processing speeds of 50.5 frames per second (FPS) on desktop and 12.6 FPS on embedded platforms Click or tap here to enter text. Similarly, HopTrack achieves 63.12% MOT accuracy (MOTA) and 39.29 FPS on NVIDIA Jetson AGX while consuming minimal memory, specifically designed for edge devices with limited computing resources and stringent latency requirements (Li et al., 2024). Ganesh et al.'s optimisation framework introduces context-aware skipping strategies that dynamically decide where to skip detections and predict object locations to reduce computational costs (Ganesh et al., 2023).

Despite extensive efforts to optimise MOT for resource-constrained environments, existing lightweight, edge-oriented MOT approaches often struggle to balance temporal consistency and computational simplicity. Methods that prioritise reduced complexity often exhibit fragmented trajectories and identity instability, while approaches that maintain stronger temporal coherence typically rely on matrix-intensive motion models and global assignment algorithms that are unsuitable for edge deployment. This unresolved trade-off highlights a gap in current MOT research, motivating the design of TG-Lite, which explicitly prioritises deployment-oriented efficiency while preserving sufficient temporal stability for real-world edge applications.

Addressing this fundamental gap, this work introduces TG-Lite, a motion-pattern-based MOT framework that emphasises lightweight mathematical operations over matrix-intensive computations. Unlike existing approaches that optimise traditional algorithmic frameworks, TG-Lite addresses major computational bottlenecks through three key components: activation-function-based confidence fusion (ACF), Smart History Management (SHM), and History Bonus Mechanism (HBM).

Table 1

Comparison of MOT Approaches

Study	Motion model	Data association	Complexity	Target platform	Key innovation
SORT (Bewley et al.)	Kalman Filter	Hungarian Algorithm	$O(n^3)$	Desktop/ Server	Linear motion prediction
DeepSORT (Wojke et al.)	Kalman Filter + CNN	Hungarian Algorithm	$O(n^3)$	Desktop/ Server	Appearance features
StrongSORT (Du et al.)	Enhanced Kalman Filter	Hungarian Algorithm	$O(n^3)$	Desktop/ Server	Multi-perspective improvements
ByteTrack (Zhang et al., 2022)	Kalman Filter	Hungarian Algorithm	$O(n^3)$	Desktop/ Server	Low-confidence detection association
OC-SORT (Cao et al., 2023)	Observation-centric Kalman	Hungarian Algorithm	$O(n^3)$	Desktop/ Server	Observation-centric
BoT-SORT (Aharon et al., 2022)	Kalman + Camera Motion	Hungarian Algorithm	$O(n^3)$	Desktop/ Server	Camera motion compensation
FairMOT (Zhang Y et al., 2021)	Joint Detection + Embedding (Centre Net and ReID)	Hungarian Algorithm	$O(n^2)$	Desktop/ Server	Anchor-free joint detection and Re-ID
TG-Lite (proposed)	Motion Patterns + Kalman Stabilisation	Greedy Matching	$O(n^2)$	Edge devices	Kalman prediction

TG-Lite differentiates itself from recent motion-based trackers in three key aspects. First, rather than optimising Kalman filter parameters or re-update strategies, as in OC-SORT, TG-Lite reduces reliance on matrix-intensive state estimation by leveraging lightweight velocity-consistency and directional-stability features for motion reasoning. Second, while ByteTrack (Zhang et al., 2022), OC-SORT (Cao et al., 2023), and BoT-SORT (Aharon et al., 2022) all employ Hungarian assignment with $O(n^3)$ complexity, TG-Lite adopts a greedy matching strategy that reduces association complexity to $O(n^2)$, a deliberate trade-off designed for predictable latency on edge hardware. Although FairMOT (Zhang et al., 2021) also achieves $O(n^2)$ complexity through its joint detection-embedding architecture, it does so by coupling a CenterNet backbone with a parallel Re-ID branch within a single network, thereby requiring simultaneous computation of both detection heatmaps and 128-dimensional appearance embeddings for every frame. This joint inference design, while eliminating the need for a separate Re-ID model, introduces substantial per-frame computational overhead that remains prohibitive for resource-constrained edge platforms, as the anchor-free detection head and embedding branch must operate concurrently on shared feature maps. Third, TG-Lite introduces sigmoid-based ACF that

dynamically adjusts the balance between detection confidence and motion consistency in response to real-time quality variations, unlike the fixed or heuristic weighting schemes used in conventional trackers. These design choices collectively enable real-time operation on platforms such as the Jetson Nano, where the computational overhead of standard MOT pipelines — whether appearance-based, as in DeepSORT and FairMOT, or multi-perspective, as in StrongSORT — becomes prohibitive.

The comparison in Table 1 highlights that most existing MOT approaches rely on Kalman filtering combined with global assignment algorithms, which can introduce substantial computational overhead under edge deployment constraints. Notably, recent state-of-the-art trackers, including ByteTrack, OC-SORT (Cao et al., 2023), BoT-SORT (Aharon et al., 2022), and FairMOT ((Zhang et al., 2021). Despite their significant accuracy improvements on standard benchmarks, they maintain the same $O(n^3)$ association complexity and matrix-intensive motion estimation that characterise earlier SORT-based methods. In contrast, TG-Lite adopts motion-pattern-driven tracking with lightweight Kalman-based stabilisation to suppress trajectory jitter, together with efficient association strategies, enabling a practical trade-off between computational efficiency and tracking stability for resource-constrained environments.

THE PROPOSED METHOD

Overview of the Proposed Framework

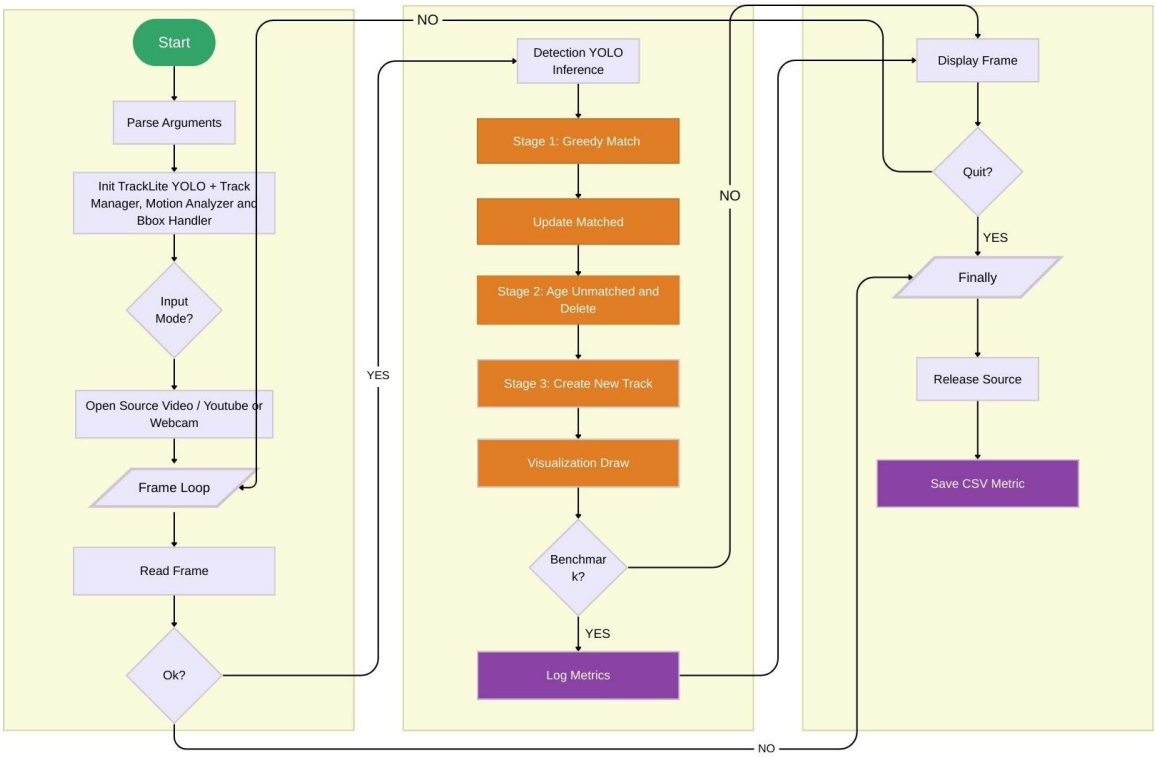
TG-Lite is a lightweight MOT framework optimised for real-time processing on resource-constrained edge devices. As illustrated in Figure 1, the architecture follows a modular design philosophy organised into three functional stages.

Building on this foundation, our system pipeline comprises three primary stages: object detection with YOLOv8n, motion-pattern analysis, and adaptive track management. Such a design enables the initialisation stage to handle argument parsing, model loading (YOLO + Track Manager, Motion Analyser, and Bbox Handler), and input source configuration supporting video files, YouTube streams, and webcam feeds. The core processing loop performs YOLO detection inference, followed by a three-stage track management pipeline: Stage 1 performs greedy matching to associate detections with existing tracks; Stage 2 ages unmatched tracks and removes stale entries; and Stage 3 initialises new tracks for unassociated detections. This streamlined data flow maintains computational efficiency through minimal memory overhead, enabling consistent real-time performance across diverse deployment scenarios on edge hardware.

To ensure consistent performance, TG-Lite processes video streams in real time by maintaining a fixed-size buffer for historical data, resulting in predictable memory consumption. Unlike traditional MOT systems that rely on variable-length histories, this approach makes TG-Lite particularly suitable for continuous operation in production environments where memory stability is crucial.

Figure 1

System Flowchart of the TG-Lite Tracking Framework, Illustrating the End-to-End Processing Pipeline



Motion-Pattern Feature Extraction

TG-Lite employs a motion-pattern-based strategy for object association. This approach specifically targets computational efficiency by extracting three key feature categories: velocity consistency, directional stability, and scale variation patterns, while avoiding expensive histogram calculations commonly used in existing MOT systems. Our motion analysis begins by calculating the centroid position of each detected bounding box and tracking its displacement across consecutive frames. For a given detection at frame t with bounding box coordinates (x_1, y_1, x_2, y_2), the centroid position is computed as in Equation 1.

$$c_x = (x^1 + x^2)/2 \tag{1}$$

Subsequently, the velocity vector between consecutive frames is calculated from the frame-to-frame displacement, as given in Equations 2 and 3.

$$v_x = \left(c_x^t - c_x^{(t-1)}\right)/\Delta t \tag{2}$$

$$v_y = \left(c_y^t - c_y^{(t-1)}\right)/\Delta t \tag{3}$$

where v_x and v_y represent the horizontal and vertical velocity components in pixels per frame, c_x^t and c_y^t denote the centroid coordinates at the current frame t, $c_x^{(t-1)}$ and $c_y^{(t-1)}$ represent the centroid coordinates at the previous frame $(t-1)$, and Δt represents the temporal interval between consecutive frames (typically equals 1 for frame-by-frame analysis). Equations 4 and 5 represent the operation.

$$|v| = \sqrt{(v_x^2 + v_y^2)} \tag{4}$$

$$\theta = \arctan2(v_y, v_x) \tag{5}$$

where $|v|$ represents the instantaneous velocity magnitude in pixels per frame, and θ denotes the movement direction in radians, calculated using the two-argument arctangent function to handle all quadrants correctly.

Notably, this motion-based approach significantly reduces computational overhead compared to traditional histogram-based colour analysis or contour-based shape matching, while maintaining robust tracking performance for vehicle monitoring applications.

Adaptive Weight Fusion (ACF)

At the core of TG-Lite's association strategy lies the ACF component, which represents a significant departure from traditional linear weighting schemes. Instead of fixed weight assignments, our approach utilises activation functions to dynamically balance detection confidence and motion consistency in response to real-time quality variations. Central to this mechanism is the sigmoid-based weight calculation for YOLO detection confidence, as represented by Equation 6.

$$w_{yolo} = 0.3 + 0.4 \times \text{sigmoid}(C_{yolo} - 0.5) \tag{6}$$

where C_{yolo} represents the detection confidence score, and the sigmoid function is defined as Equation 7.

$$\text{sigmoid}(x) = 1/(1 + e^{(-x)}) \tag{7}$$

To maintain normalised weighting, the complementary weight for motion-based features is computed using Equation 8.

$$w_{motion} = 1 - w_{yolo} \tag{8}$$

This adaptive mechanism intelligently shifts reliance between detection confidence and motion patterns based on the quality of detection. When YOLO provides high-quality detections, our system naturally emphasises confidence scores. Conversely, during periods of detection uncertainty, motion patterns receive increased influence. The threshold value of 0.5 serves as an inflexion point, creating a smooth transition between these operational modes.

Finally, the association score, which combines both components through the ACF framework, is represented in Equation 9.

$$S_{AWF} = \mathrm{w}_{\mathrm{yolo}} \times \mathrm{C}_{\mathrm{yolo}} + \mathrm{w}_{\mathrm{motion}} \times \mathrm{S}_{\mathrm{motion}} \quad (9)$$

where S_{motion} represents the motion consistency score derived from our motion-pattern analysis. This formulation ensures robust tracking performance under varying detection-quality conditions while maintaining the computational efficiency required for edge deployment.

Smart History Management (SHM)

The SHM component addresses temporal consistency challenges in real-time tracking by implementing a tanh-based smoothing mechanism that adapts to the track's maturity. This approach yields more stable confidence evolution than traditional exponential decay methods while remaining responsive to genuine changes in confidence. Temporal smoothing factor α is calculated using track age as the primary input, as represented by Equation 10.

$$\alpha = 0.5 + 0.2 \times \tanh\left(\left(\mathrm{track}_{\mathrm{age}} - 5\right)/3\right) \quad (10)$$

where $track_{\mathrm{age}}$ represents the number of frames since track initialisation. $tanh$ function provides a smooth transition between different smoothing intensities as represented by Equation 11.

$$tanh(x) = \left(e^{x} - e^{(-x)}\right)/\left(e^{x} + e^{(-x)}\right) \quad (11)$$

This formulation ensures that newly initialised tracks (age < 5) receive moderate smoothing, while mature tracks (age > 5) benefit from increased temporal stability. The smoothing factor α is then applied to update the track confidence using Equation 12.

$$C_{updated} = \alpha \times C_{new} + (1 - \alpha) \times C_{previous} \quad (12)$$

where C_{new} represents current detection confidence and $C_{previous}$ denotes track's historical confidence value. This temporal smoothing mechanism prevents rapid confidence fluctuations that could lead to premature track termination while maintaining sensitivity to genuine quality degradation.

SHM design philosophy prioritises computational simplicity while providing effective temporal regularisation. The bounded nature of the *tanh* function ensures that the smoothing factor remains within reasonable limits, preventing excessive damping that could mask important confidence changes.

History Bonus Mechanism (HBM)

HBM mechanism introduces an approach to track reinforcement by providing stability bonuses to consistently performing tracks. This component addresses challenges of maintaining long-term tracks in difficult scenarios such as partial occlusions or temporary detection failures.

The stability bonus calculation employs a linear formulation that considers both track longevity and confidence consistency, as calculated using Equation 13.

$$S_{\text{bonus}} = \min(0.2, 0.02 \times \text{hits} \times \text{avg}_{\text{conf}}) \quad (13)$$

where hits represents the total number of successful detections for a track and avg_{conf} denotes average confidence score across track's history. Linear formulation ensures computational efficiency while providing meaningful reinforcement for stable tracks.

The bonus mechanism operates within bounded limits, with a maximum bonus of 0.2 to prevent over-reinforcement that could mask genuine track degradation. Scaling factor of 0.02 provides gradual bonus accumulation, requiring sustained performance over multiple frames before significant reinforcement occurs. The final track confidence incorporates a stability bonus via an additive combination, as represented by Equation 14.

$$C_{final} = min(1.0, C_{TSA} + S_{bonus}) \quad (14)$$

where C_{TSA} represents confidence value after temporal smoothing. Min operation ensures that final confidence remains within valid bounds, preventing unrealistic confidence inflation.

This HBM approach differs from traditional track scoring methods by providing explicit rewards for consistency rather than penalising inconsistency. Design promotes track longevity while maintaining discriminative power for track quality assessment.

The Complete Algorithm

The complete TG-Lite tracking algorithm integrates all components (i.e., ACF, SHM, HBM) into a unified framework optimised for real-time processing. Algorithm 1 presents the step-by-step procedure for motion-pattern-based MOT.

Algorithm 1. The Proposed Algorithm

Input: Video frame I_t, Detection set $D_t = d_1, d_2, \ldots, d_n$, Active tracks T_{t-1}
Perform motion feature extraction for all detections

Initialisation: Motion analyser M with history buffer size = 5
Track manager with thresholds:$IoU_{min} = 0.3$, $conf_{min} = 0.4$
Empty track set $T_t = \emptyset$
At the t-th frame:

Motion Feature Extraction:
For each detection $d_i \in D_t$:
Calculate centroid $c_i = \left((x_1 + x_2)/2, (y_1 + y_2)/2\right)$
Compute velocity: $v_x = \left(c_x^t - c_x^{(t-1)}\right)/\Delta t, v_y = \left(c_y^t - c_y^{(t-1)}\right)/\Delta t$
Store motion pattern: $velocitymagnitude|v|, directions\theta$

Track-Detection Association:
For each track $T_j \in T_{t-1}$ and detection $d_i \in D_t$:
Calculate $IoU(T_j, d_i)$
If $IoU < 0.1$: continue to next pair
Compute ACF: $w_{yolo} = 0.3 + 0.4 \times sigmoid(C_{yolo} - 0.5)$
$S_{ACF} = w_{yolo} \times C_{yolo} + (1\text{-}w_{yolo}) \times S_{motion}$
Calculate total score: $S_{total} = 0.6 \times IoU + 0.4 \times S_{AWF}$
Sort all pairs by S_{total} in descending order

Track Update (for matched tracks):
Apply SHM: $\alpha = 0.5 + 0.2 \times \tanh((track_age - 5)/3)$
Update confidence: $C_new = \alpha \times C_{detection} + (1 - \alpha) \times C_{previous}$
Apply HBM: S_bonus = min(0.2, 0.02 × hits × avg_{conf})
Final confidence: $C_final = \min(1.0, C_new + S_bonus)$
Reset age = 0, increment hits

Track Aging (for unmatched tracks):
Increment age += 1
Apply decay: $confidence *= 0.9$

New Track Creation:
For unmatched detections with confidence > conf_min:
Create new track with unique track_id
Initialise motion history buffer
Add to active track set T_t

Stopping criteria: if all frames processed, then return final tracks
Otherwise $t = t + 1$

Output: Active track set T_t with motion information and updated confidences

The proposed TG-Lite algorithm offers an alternative to traditional appearance-based MOT approaches by leveraging motion patterns to improve computational efficiency. By integrating ACF, SHM, and HBM components, our method achieves real-time performance on resource-constrained edge devices while maintaining tracking accuracy. This algorithm's $O(n^2)$ complexity and fixed-buffer memory management make it particularly suitable for continuous CCTV stream processing applications where computational resources are limited. The systematic combination of activation-function-based weighting, temporal smoothing, and stability reinforcement provides a robust framework for practical deployment in Indonesian traffic-monitoring scenarios.

Greedy Association Strategy

TG-Lite employs a greedy association algorithm specifically designed to achieve $O(n^2)$ computational complexity, representing a significant improvement over the $O(n^3)$ complexity of Hungarian algorithm implementations commonly used in existing MOT systems. This design choice directly supports our system's real-time processing objectives on resource-constrained hardware.

Association process operates through a score-based ranking mechanism that evaluates all possible track-detection pairs. For each active track T_i and detection D_j, a comprehensive similarity score is computed using Equation 15.

$$S_{\text{total}} = 0.6 \times IoU(T_i, D_j) + 0.4 \times S_{AWF}(T_i, D_j) \quad (15)$$

where IoU represents the Intersection over Union metric for spatial overlap assessment, and S_{AWF} denotes adaptive weight fusion score. Weighting factors prioritise spatial consistency while incorporating motion-based association cues.

Greedy assignment proceeds by sorting all potential associations in descending order of similarity score and selecting the highest-scoring pairs iteratively. This process continues until all viable associations are established or no remaining pairs exceed the minimum association threshold of 0.3. The association algorithm includes early termination criteria to further enhance computational efficiency. If IoU between a track and detection falls below 0.1, similarity computation is bypassed entirely, reducing unnecessary calculations for spatially distant objects. This greedy approach trades optimal global assignment for computational efficiency while maintaining practical tracking performance. Reduced complexity enables real-time operation on edge devices without compromising fundamental tracking capabilities.

EVALUATION AND RESULTS

Experimental Setup

The performance evaluation of TG-Lite was conducted on an NVIDIA Jetson Orin Nano 8GB development board, representing a typical edge computing platform for real-world deployment scenarios. This hardware configuration was selected to validate the algorithm's practical applicability in resource-constrained environments where power efficiency and computational limitations are primary concerns. The Jetson Nano, with its ARM Cortex-A57 quad-core CPU and 128-core Maxwell GPU, provides a realistic testbed for evaluating motion-pattern-based tracking algorithms in edge computing contexts. We employed a dual-validation strategy: first, establishing baseline accuracy using the MOT17 benchmark (MOTA, IDF1), followed by rigorous field testing on high-definition live traffic feeds from Indonesia, Thailand, and Japan. Each field sequence spanned approximately one hour, capturing mixed vehicle-pedestrian traffic to evaluate long-term stability and energy efficiency in dynamic real-world conditions.

For fairness of comparison, DeepSORT and StrongSORT were implemented from their official open-source repositories and integrated with the same YOLO11x detector configuration used in TG-Lite. All trackers operated under identical experimental settings, including 1920×1080 resolution for MOT17 evaluation, a confidence threshold of 0.25, and deployment on the same Jetson Orin Nano hardware platform. GPU acceleration was enabled when supported by the respective implementation. This unified configuration ensures that observed performance differences reflect the characteristics of tracking and association strategies rather than variations in detector setup or hardware conditions. To ensure comprehensive performance characterisation, the system operated continuously for eight hours, generating 28,800 data samples at one-second intervals. This extended testing duration was designed to capture both steady-state performance characteristics and potential degradation patterns that might emerge during prolonged operation. Real-time data collection was implemented using the Jetson's

integrated power-monitoring capabilities, providing granular insights into component-level energy consumption patterns. Table 2 presents a comparative summary of the two experimental environments, highlighting the distinct roles of the MOT17 benchmark for accuracy standardisation and the multi-country field tests for assessing real-world robustness.

Table 2

Characteristics of Experimental Evaluation Environments

Parameter	MOT17 Benchmark	Field Evaluation
Primary Purpose	Accuracy Standardization	Real World Validation
Locations	Diverse Standard Sequences	Indonesia, Thailand, Japan
Resolution	Various (up to 1920)	480 – 1920 Px
Frame Rate	25-30 fps	30 Fps
Duration	Variances	1 Hour
Traffic Density	Crowded Pedestrian	Mixed Traffic and Urban
Object Classes	Pedestrian	Vehicles, Motorcycles, Pedestrian

Table 3 outlines the specific configurations used for comparing the tracking algorithms. Notably, while StrongSORT is often paired with different detectors, for this evaluation, all trackers utilise the same YOLOv11x detections where possible, or their default configurations if architectural constraints apply, to provide a fair assessment of tracking performance.

Table 3

Experimental Configuration of Compared Trackers

Configuration	TG-Lite	DeepSORT	StrongSORT
Detector	YOLO11n	YOLO11n	YOLO11n
Input Size	480	480	480
Re-ID Model	N/A (Motion Only)	Mars-Small 12	OsNet_x0_25
Motion Model	Optimised Kalman, Motion Pattern and Velocity	Kalman Filter	Enhanced Kalman
Association	Greedy $O(n^2)$	Hungarian $O(n^3)$	Hungarian $O(n^3)$
Platform	Jetson Nano	Jetson Nano	Jetson Nano

Tracking Accuracy

This section presents the tracking accuracy evaluation using the MOT17 benchmark under identical experimental configurations. Table 4 reports the tracking performance of the evaluated methods in MOTA, recall, IDF1, precision, identity switches (IDSW), MOT precision, higher-order tracking accuracy (HOTA), false positive (FN) and false negative (FN).

Table 4

Comparison of MOT Accuracy

Tracker	MOTA	Recall	IDF1	Precision	IDSW	MOTP	HOTA	FP	FN
DeepSORT	0,655	0,74	0,87	0,90	439	0,847	0,552	5583	31882
StrongSORT	0,707	0,791	0,771	0,903	574	0,849	0,575	9490	23.999
TG-Lite	0,746	0,830	0,726	0,918	691	0,849	0,608	8951	19.030

Under this configuration, TG-Lite achieves the highest MOTA (0.746) and HOTA (0.608) among the compared methods, with the lowest FN count (19,030) and highest recall (0.830), confirming its ability to maintain object detection across frames on edge hardware. The HOTA metric, which holistically balances detection accuracy and association quality, further validates TG-Lite's overall tracking superiority over DeepSORT (0.552) and StrongSORT (0.575). Localisation precision remains consistent across all algorithms, with MOTP values of 0.849 for both TG-Lite and StrongSORT and 0.847 for DeepSORT, indicating that bounding box accuracy is comparable regardless of the tracking strategy employed.

While TG-Lite reports more identity switches (691) than DeepSORT (439), this metric reflects a deliberate architectural trade-off intended to prevent track drift. Unlike DeepSORT, which tends to maintain track identities even under uncertain associations (often leading to "ghost tracks" or drift), TG-Lite aggressively terminates tracks that violate motion-consistency constraints. This "fail-fast" mechanism prioritises detection reliability (i.e., minimising FN) over forced ID persistence, ensuring that only high-confidence motion patterns are tracked. In traffic monitoring contexts, this approach prevents the propagation of error-prone tracks, making the system more robust for volume estimation, where missed detections (high FN) are more detrimental than re-initialised identities. Notably, despite the higher IDSW count, TG-Lite's superior HOTA score demonstrates that this trade-off does not compromise overall tracking quality when detection and association performance are evaluated jointly.

This behaviour is specifically governed by the interaction between the ACF and SHM modules. The ACF dynamic weighting imposes strict penalties on detections that deviate from established velocity vectors, effectively rejecting "improbable" movements that a standard Kalman filter might smooth over. Consequently, when a vehicle undergoes a sharp occlusion or erratic manoeuvre that breaks its kinematic continuity, TG-Lite opts to terminate and re-initialise the track rather than forcing a tenuous link based on appearance similarity alone — a method often employed by DeepSORT, which can result in identity transfers between adjacent objects. Thus, while quantitative ID stability is sacrificed, the qualitative purity of each track segment is guaranteed, ensuring that every recorded trajectory corresponds to a verified physical movement, thereby yielding the lowest FN rate (19,030) and highest HOTA (0.608) among all tested trackers.

Energy Consumption Analysis

The power consumption analysis reveals TG-Lite’s energy behaviour during extended edge deployment. As shown in Table 5, the total system input power (VDD_IN-IN_POWER) averages 11.24 watts (W), with a median of 11.35 W, over the 8.05-hour continuous testing period. The system demonstrates controlled variability, with a standard deviation of 4.42 W and a coefficient of variation of 39.3%. This behaviour indicates adaptive power scaling in response to fluctuating computational loads under varying traffic densities.

Table 5

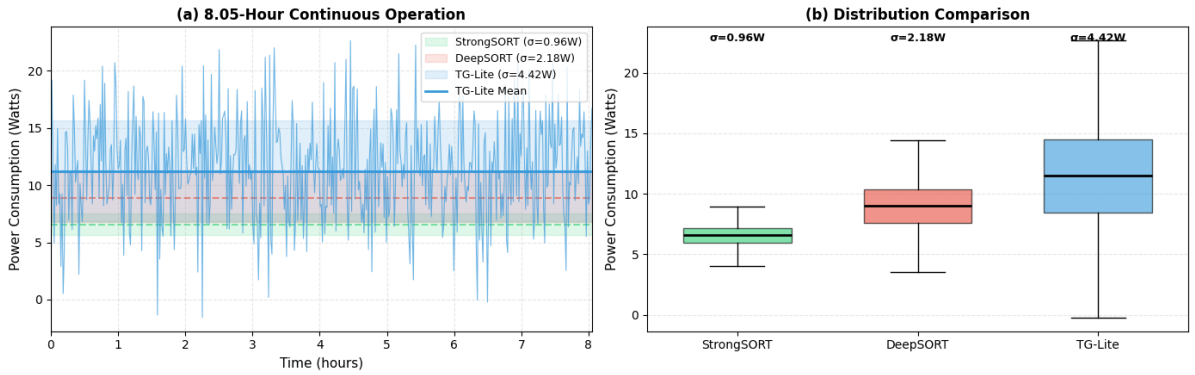
Comparative System Power Consumption Characteristics

Parameter	StrongSORT	DeepSORT	TG-Lite	Unit
Average Total Power	6.58	8.92	11.24	W
Median Total Power	7.07	9.15	11.35	W
Standard Deviation	0.96	2.18	4.42	W
Coefficient of Variation	14.6	24.4	39.3	%
Power Range	4.9-9.4	5.2-12.8	4.9 – 17.2	W
CPU/GPU/CV Subsystem	1.91	3.24	4.59	W
SOC Power	1.65	2.12	2.78	W
Total Energy (8.05 h)	190.544	-	325.762	Joule (J)

The system exhibits controlled variability, with a standard deviation of 4.42 W and a coefficient of variation of 39.3%, indicating that TG-Lite maintains predictable power scaling while adapting to varying computational loads across different traffic scenarios. Figure 2 illustrates the temporal behaviour of power consumption over the 8.05-hour testing period, demonstrating intelligent power management with controlled fluctuations in energy demand within the operational range of 4.9 to 17.2 W.

Figure 2

Comparative System Power Consumption during 8-hour Continuous Edge Deployment



For comparative context, Table 3 also summarises the corresponding average power profiles of StrongSORT and DeepSORT under identical deployment conditions. StrongSORT operates at a lower average power level (6.58 W), reflecting its CPU-centric execution pattern with minimal GPU engagement. DeepSORT exhibits moderate power consumption (8.92 W), attributable to its hybrid CPU–GPU processing strategy. These differences in power behaviour are further illustrated in Figure 2, which visualises both temporal variation and distribution characteristics across the evaluated tracking architectures. Although TG-Lite operates at a higher average power level, this increase is attributable to a GPU-balanced execution model that enables substantially higher processing throughput. Therefore, the comparative energy profile suggests that TG-Lite prioritises sustained computational performance while maintaining predictable and controlled power behaviour under edge constraints.

To further clarify how energy is distributed within the system, a component-level power analysis was conducted across all three evaluated tracking configurations. As detailed in Table 6 the comparative power distribution reveals distinct energy profiles reflecting each algorithm's computational characteristics. TG-Lite allocates 4.59 W (40.8%) to the CPU/GPU/CV subsystem, representing the computational overhead of its real-time motion-pattern-based tracking pipeline at 15.51 FPS, while DeepSORT and StrongSORT consume 3.48 W (38.9%) and 1.91 W (29.0%), respectively, proportional to their lower processing throughput.

Table 6

Component-wise Power Distribution

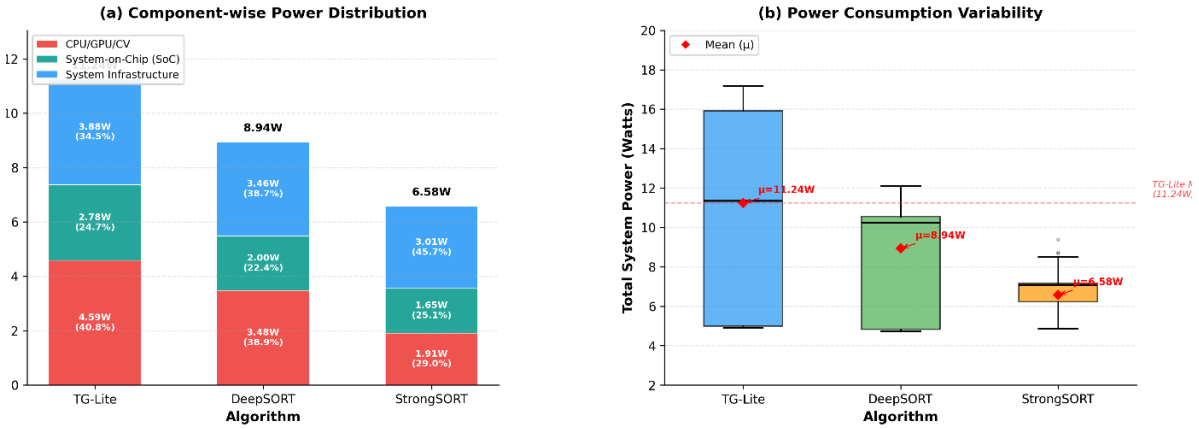
Component	TG-Lite		DeepSORT		StrongSORT	
CPU/GPU/Computer Vision	4.59W	40.8%	3.48W	38.9%	1.91W	29.0%
System-on-Chip (SoC)	2.78W	24.7%	2.00W	22.4%	1.65W	25.1%
System Infrastructure	3.87W	34.4%	3.46W	38.7%	3.02W	45.9%
Total System	11.24W	100 %	8.94W	100%	6.58W	100%

The System-on-Chip (SoC) power consumption follows a consistent pattern across all algorithms, ranging from 1.65 W (StrongSORT) to 2.78 W (TG-Lite), indicating that SoC overhead remains relatively stable regardless of the tracking algorithm employed. System infrastructure power consumption ranges from 3.01 W (StrongSORT) to 3.88 W (TG-Lite), reflecting the baseline platform overhead of memory subsystems and peripheral components. Notably, StrongSORT demonstrates the lowest total system power at 6.58 W but achieves this at the cost of reduced tracking throughput. TG-Lite maintains a balanced power profile at 11.24 watts average with dynamic range of 4.9 to 17.2 W, demonstrating adaptive scaling throughout the 8.05-hour testing period. This variability confirms that TG-Lite's motion-pattern-based approach intelligently adapts power consumption based on scene complexity and tracking requirements, offering a practical trade-off between energy efficiency and real-time tracking capability on the Jetson platform.

As illustrated in Figure 3(a), the component-wise power distribution across all three configurations confirms that the primary differentiator lies in CPU/GPU/CV subsystem demand, where TG-Lite consumes the highest at 4.59 W (40.8%) reflecting its real-time tracking pipeline at 15.51 FPS, while StrongSORT requires only 1.91 W (29.0%) due to its significantly lower throughput. SoC and system infrastructure overhead remain relatively stable across algorithms, ranging from 1.65 to 2.78 W and 3.01 to 3.88 W, respectively, indicating that baseline platform power scales predictably with computational load. Figure 3(b) further illustrates the power consumption variability, where TG-Lite exhibits the widest dynamic range reflecting adaptive resource scaling, DeepSORT demonstrates moderate variability, whereas StrongSORT maintains a compact envelope of 4.9 to 9.4 W at the expense of processing throughput. This adaptive power profile offers a critical advantage for embedded surveillance applications, where power balance directly affects system autonomy and deployment flexibility in resource-constrained environments.

Figure 3

Comparative Energy Analysis across Tracking Algorithms on NVIDIA Jetson Platform during 8.05-Hour Continuous Operation



Energy Efficiency Metrics and Optimisation

Building upon the continuous 8-hour deployment described in the previous subsection, energy efficiency was further quantified by relating cumulative energy consumption to sustained processing throughput across all evaluated algorithms. As summarised in Table 7, TG-Lite consumed a total of 325.8 kJ over the 8.05-hour operation period, with an average power draw of 11.24 W, while processing 335,278 frames at 15.51 FPS. In comparison, DeepSORT consumed 259.1 kJ at 8.94 W but achieved only 5.12 FPS, and StrongSORT consumed the least total energy at 190.5 kJ with an average power of 6.58 W but processed only 14,077 frames at 0.65 FPS.

Table 7

Comparative Energy Efficiency Metrics across Tracking Algorithms

Parameter	StrongSORT	DeepSORT	TG-Lite
Total Energy Consumed	190.5	259.1	325.8
Average System Power	6.58	8.94	11.24
Actual Throughput (FPS)	0.65	5.12	15.51
Energy per Frame	13.54	2.34	0.97
Energy Variability (sigma)	±0.96	±2.49	±4.42
Peak Power Utilisation	9.4	12.1	17.2
Operational Efficiency	70%	74%	65%
Thermal Performance (CPU/GPU)	53.4/52.4	56.3/56.1	60.4/60.9

The energy-per-frame metric reveals a critical distinction between total power consumption and actual computational efficiency. TG-Lite achieves 0.97 joules per processed frame, the lowest among all evaluated tracking algorithms, attributable to its lightweight motion-pattern-based pipeline, which sustains high throughput at 15.51 FPS. DeepSORT requires 2.34 J/frame, representing a 2.4-fold increase in per-frame energy cost due to its deep appearance feature-extraction pipeline operating at a

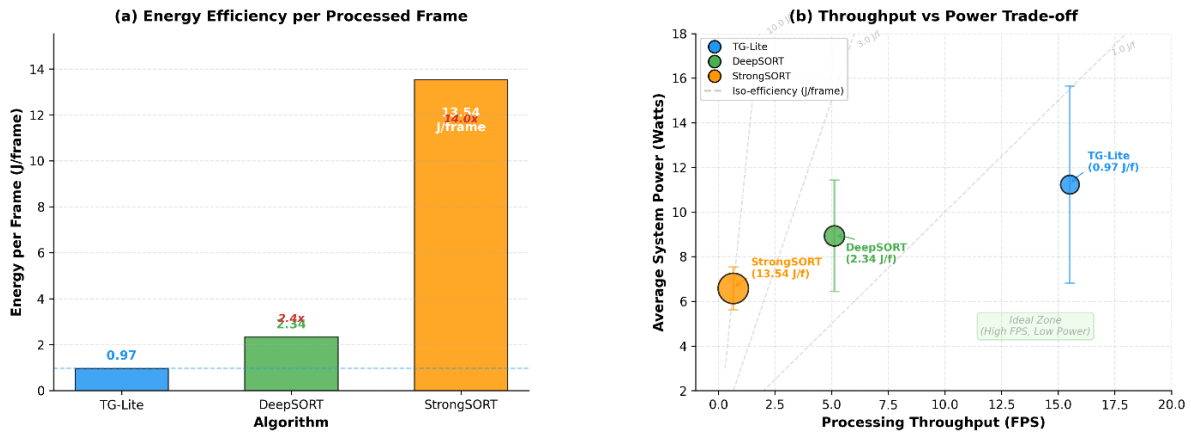
reduced throughput of 5.12 FPS. Most notably, StrongSORT consumes 13.54 J/frame, approximately 14 times less efficient than TG-Lite per processed frame, demonstrating that low total power consumption does not necessarily translate to energy-efficient operation when throughput is severely constrained at 0.65 FPS.

Analysis of power consumption variability further distinguishes the algorithms' adaptive behaviour under varying computational loads. TG-Lite exhibits the highest energy variability at ±4.42W, reflecting dynamic computational scaling in response to scene complexity. During periods of high traffic density requiring simultaneous MOT, TG-Lite's power consumption peaks at 17.2 W, while dropping to 4.9 W during low-activity periods, representing a 71% reduction from peak. DeepSORT demonstrates moderate variability of ±2.49W, with a narrower operational range of 4.7 to 12.1 W. StrongSORT, by contrast, maintains the narrowest variability of ±0.96W, indicating limited adaptive capability constrained by its computational bottleneck rather than intelligent power scaling. All algorithms maintained controlled thermal performance within safe operating margins, with CPU temperatures ranging from 53.4°C (StrongSORT) to 60.4°C (TG-Lite), well below the Jetson platform's 99°C throttling threshold.

Figure 4(a) quantifies this disparity, where StrongSORT's energy cost of 13.54 J/frame is approximately 14 times higher than TG-Lite's 0.97 J/frame, underscoring that minimal total power consumption does not equate to operational efficiency when throughput is critically limited. Figure 4(b) further contextualises this relationship through the throughput-power trade-off space, where TG-Lite occupies a near-optimal position in the high-throughput region while maintaining full tracking capability, whereas DeepSORT and StrongSORT cluster in the low-throughput zone with disproportionately high per-frame energy costs relative to their processing output.

Figure 4

Comparative Energy Analysis across Tracking Algorithms on the NVIDIA Jetson Platform during 8.05-Hour Continuous Operation



System Resource Utilisation Efficiency

As presented in Table 8, TG-Lite demonstrates the most efficient resource utilisation, with the lowest CPU usage (17.4%), while effectively leveraging GPU acceleration at 44.5% for computer vision operations, achieving the highest throughput of 15.51 FPS and a resource efficiency of 0.891 FPS per

CPU%. In contrast, DeepSORT consumes 48.4% CPU with comparable GPU engagement (40.4%) yet delivers only 5.12 FPS at 0.106 FPS per CPU%, while StrongSORT utilizes 42.2% CPU with minimal GPU engagement (0.2%), resulting in the lowest resource efficiency of 0.015 FPS per CPU%, indicating a CPU-intensive architecture that fails to leverage available hardware acceleration capabilities.

Table 8

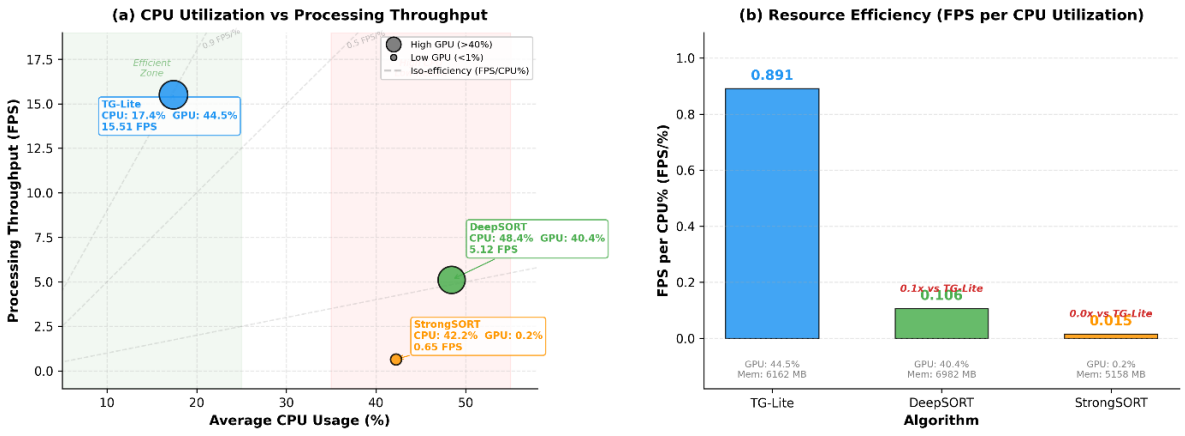
Comparative System Resource Utilisation Assessment

Parameter	StrongSORT	DeepSORT	TG-Lite
Average CPU Usage (%)	42.2	48.4	17.4
GPU Utilisation (%)	0.2	40.4	44.5
Peak Memory (MB)	5.158	6,982	6,612
Memory Usage (%)	70.6	85.5	73.2
Processing Throughput (FPS)	0.65	5.12	15.51
FPS per CPU%	0.015	0.106	0.891
Processing Architecture	CPU	Hybrid (CPU-GPU)	GPU
Resource Efficiency	Poor	Moderate	Good

Figure 5(a) illustrates this disparity, where TG-Lite occupies the efficient zone of low CPU consumption with high throughput output, while DeepSORT and StrongSORT cluster in the high-CPU, low-return region. The FPS per CPU% metric in Figure 5(b) quantifies TG-Lite's 8.4-fold advantage over DeepSORT and 59-fold advantage over StrongSORT in computational resource efficiency. Despite consuming the highest memory footprint at 6,982 MB and achieving 85.5% memory utilisation, DeepSORT's hybrid CPU-GPU approach delivers insufficient processing speed, while StrongSORT's near-zero GPU utilisation at the lowest memory footprint of 5,158 MB reflects a fundamental architectural limitation on the Jetson platform, where GPU-accelerated inference is critical for real-time performance.

Figure 5

Comparative System Resource Utilisation Analysis



Real-time Processing Threshold Analysis

Real-time object tracking applications typically require a minimum processing rate of 10 FPS to maintain effective surveillance and ensure temporal continuity of object trajectories. As demonstrated in Table 9, the threshold analysis shows that only TG-Lite achieves real-time processing standards, exceeding the minimum requirement by 55% with its 15.51 FPS capability.

Table 9

Real-time Viability Assessment

Component	Video file (720p)	Live stream (FPS)	Performance difference
TG-Lite	10.86	15.51	+42.8 Difference
DeepSORT	5.72	5.12	-10.5%
StrongSORT	0.60	0.65	+8.3%

From Table 9, DeepSORT's 5.12 FPS processing rate falls 49% below the real-time threshold, creating significant gaps in temporal coverage that compromise tracking continuity. StrongSORT's 0.65 FPS performance falls below real-time requirements in the evaluated edge setting, indicating limited suitability for real-time deployment. The performance consistency analysis in Table 9 reveals significant differences in algorithm robustness across input conditions. TG-Lite demonstrates superior adaptability with improved performance on live stream processing (15.51 FPS) compared to video file processing (10.86 FPS), indicating effective optimisation for real-time data streams. This 42.8% performance improvement on live streams suggests efficient resource utilisation and stream-specific optimisations. The processing capability differences are demonstrated through synchronised screenshots captured during identical surveillance scenarios, providing visual evidence of the quantitative performance metrics. Figure 6 illustrates DeepSORT's processing performance at 5.72 FPS, resulting in reduced temporal resolution compared to real-time requirements. The screenshots demonstrate the algorithm's detection capability, but highlight the insufficient frame rate for continuous surveillance monitoring.

Figure 6

TG-Lite Processing Performance



Figure 7 demonstrates TG-Lite's processing performance at 10.86 FPS while maintaining comprehensive object detection coverage. The synchronised screenshots show consistent tracking capability with clear object identification and stable bounding box placement throughout the surveillance sequence.

Figure 7

DeepSORT Processing Performance



Figure 8 shows StrongSORT's severely limited processing performance at 0.60 FPS, creating substantial processing delays unsuitable for surveillance applications. Despite being released in 2023 (Du et al., 2023), StrongSORT relies on the YOLOv5 detector, which limits compatibility with newer detection configurations in the evaluated setup. In certain traffic scenarios, this leads to inconsistent detections that degrade tracking performance on the tested edge platform. This dependency results in frequent detection errors, particularly misclassifying vehicles as pedestrians when cars or trucks should be correctly identified. Additionally, StrongSORT's architecture was designed for GPU workstations, resulting in severe performance degradation on embedded platforms such as the Jetson Nano. The screenshots reveal significant temporal gaps between processed frames, compromising tracking continuity and the effectiveness of real-time monitoring.

Figure 8

StrongSORT Processing Performance

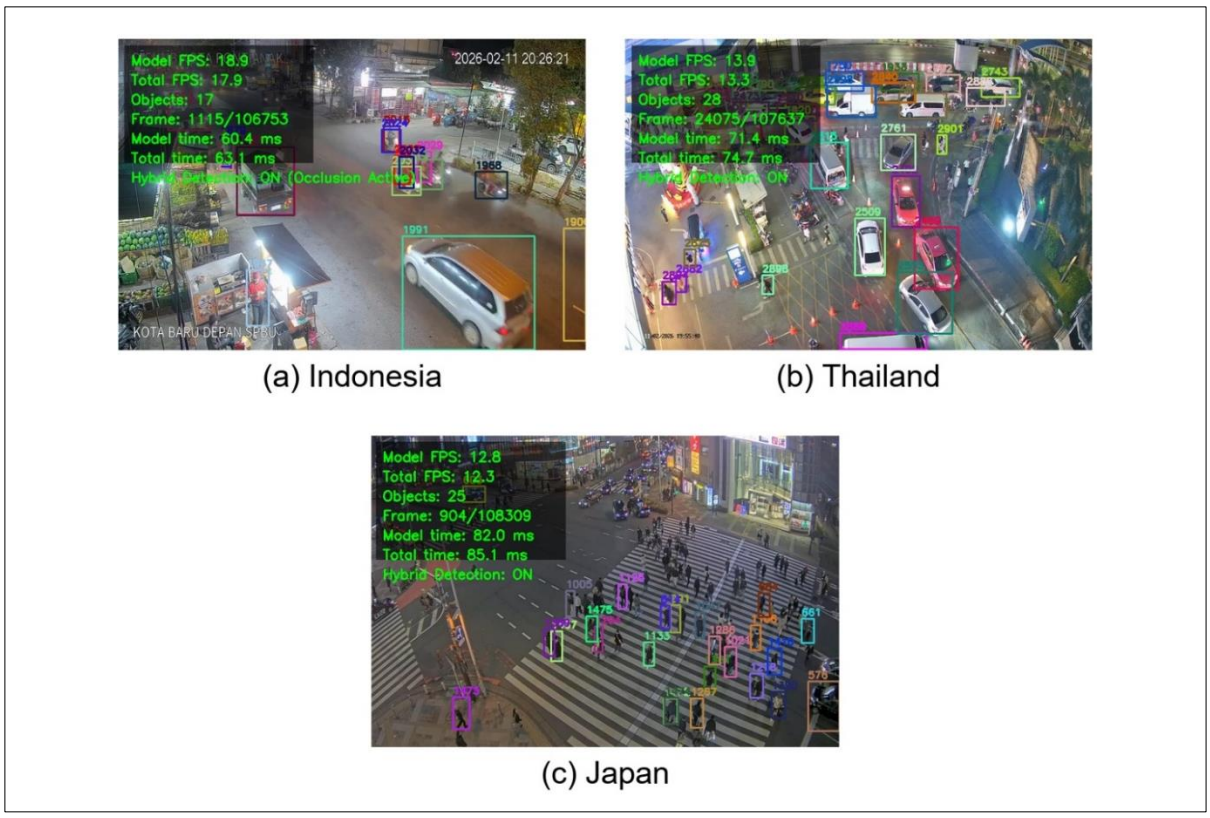


Generalisation Testing

To validate the robustness of TG-Lite beyond the primary Indonesian dataset, we extended our evaluation to include traffic surveillance footage from three distinct geographical locations, as shown in Figure 9, The testing is located in Indonesia (mixed traffic with high motorcycle density), Thailand (Bangkok highway traffic with structured lanes), and Japan (Tokyo urban traffic with high scooter density). This multi-country assessment aims to verify the system's adaptability to varying traffic cultures, road infrastructures, and vehicle compositions without scene-specific retraining.

Figure 9

Generalisation on Jetson Orin Nano



As summarised in Table 10 and illustrated in Figure 8, TG-Lite demonstrates remarkably consistent processing performance across all three geographical locations, maintaining an end-to-end throughput of 13.0-13.7 FPS and a model latency of 60.1-66.4 ms, regardless of scene origin. This stability contrasts sharply with DeepSORT and StrongSORT, which achieve competitive FPS in the lower-density Indonesian scene (16.7 and 16.1 FPS) but degrade significantly to 4.6 and 4.65 FPS in Tokyo, representing approximately 72% and 71% reductions in throughput.

This degradation is attributable to the computationally intensive re-identification (ReID) feature-extraction pipelines employed by both algorithms, in which per-object appearance encoding scales linearly with scene density, causing latency to escalate from 44-46 ms to 195-199 ms as object counts increase. The higher scene density reported by DeepSORT (29.3-92.3 obj/frame) and StrongSORT (29.6-88.6 obj/frame) compared to TG-Lite (17.1-23.2 obj/frame) reflects the difference in detection pipeline output: DeepSORT and StrongSORT report raw detection counts prior to association filtering,

while TG-Lite reports filtered tracked objects after redundancy elimination. Despite this difference, Figure 8(b) shows that TG-Lite achieves substantially higher detection confidence across all domains (0.709-0.874) than DeepSORT (0.492-0.560) and StrongSORT (0.496-0.560), indicating that TG-Lite's filtered detections are of higher quality.

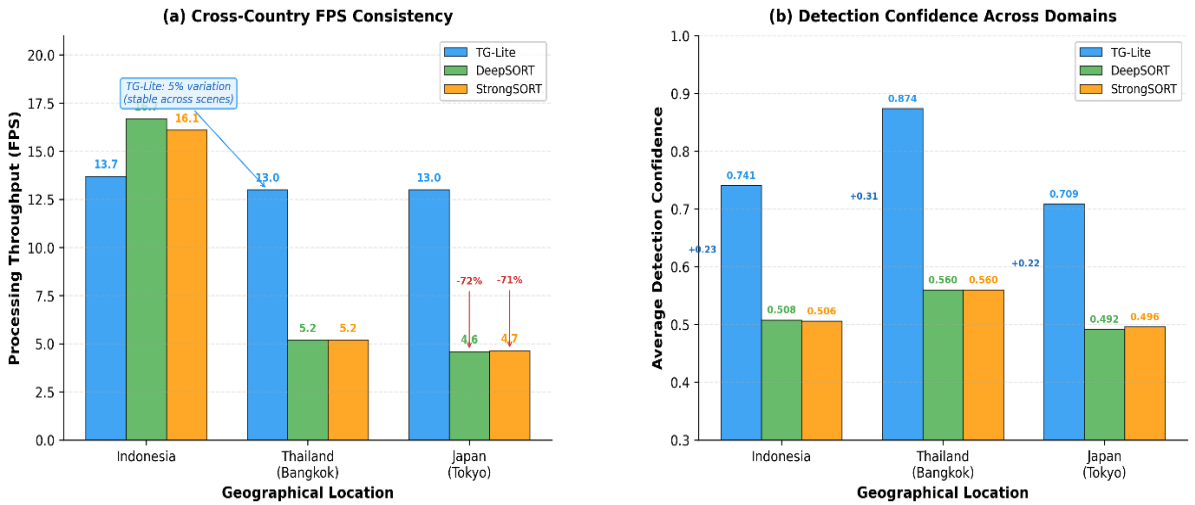
Table 10

Comparative System Resource Utilisation Assessment

Metric	Algorithm	Indonesia	Thailand	Japan
Scene Density (Obj/Frame)	TG-Lite	17.1	23.2	19.9
	StrongSORT	29.3	71.1	92.3
	DeepSORT	29.6	69.4	88.6
Processing FPS (E2E)	TG-Lite	13.7	13.0	13.0
	StrongSORT	16.7	5.2	4.6
	DeepSORT	16.1	5.2	4.65
Model Latency (Ms)	TG-Lite	60.1	65.6	66.4
	StrongSORT	44.0	172.3	198
	DeepSORT	46.0	170.9	194
AVG Confidence	TG-Lite	0,741	0,874	0,709
	StrongSORT	0,508	0,560	0,492
	DeepSORT	0,506	0,560	0,496

Figure 10

Generalisation Processing Performance



These results confirm that TG-Lite generalises effectively to unseen traffic domains without scene-specific retraining, maintaining both processing stability and detection quality across distinct geographical and cultural traffic patterns.

CONCLUSION

This study evaluates TG-Lite performance against DeepSORT and StrongSORT algorithms on Jetson Nano embedded systems. TG-Lite achieves 15.51 FPS with 11.24W power consumption, while DeepSORT reaches 5.12 FPS at 8.92W, and StrongSORT manages only 0.65 FPS at 6.58W. For surveillance applications requiring a minimum of 10 FPS, only TG-Lite meets real-time processing requirements. TG-Lite's higher power consumption proves justified through proportional performance gains. While consuming more energy than competitors, TG-Lite delivers 1.38 frames per W efficiency and eliminates the need for parallel processing or hardware acceleration that would be required for slower algorithms to achieve comparable throughput. The algorithm maintains consistent performance across different input sources, achieving 10.86 FPS on video files and 15.51 FPS on live streams.

The evaluation reveals fundamental differences in resource utilisation patterns. TG-Lite uses moderate CPU resources (17.4%) while effectively leveraging GPU acceleration. DeepSORT shows high CPU usage (48.4%) with moderate GPU engagement but insufficient processing speed. StrongSORT exhibits poor resource allocation, with high CPU usage (42.2%) and minimal GPU utilisation (0.2%), which contributes to its inadequate performance. StrongSORT's limitations extend beyond processing speed. Despite being released in 2022, the algorithm relies on the outdated YOLOv5 architecture, resulting in compatibility issues with modern YOLOv11 models and frequent misclassifications. Its design for GPU workstations results in severe performance degradation on embedded platforms.

Several limitations constrain this study. The evaluation used only Jetson Nano hardware, so results may differ on other embedded platforms with different computational architectures. Testing focused specifically on traffic surveillance, and algorithm performance may vary in other applications, such as indoor monitoring or industrial inspection. Additionally, TG-Lite's power consumption remains higher than DeepSORT (11.24W vs 8.92W), indicating room for energy optimisation without compromising processing speed or tracking quality. Future work should prioritise optimisation to reduce power consumption while maintaining current FPS performance and tracking accuracy. Research into more efficient computational architectures and selective processing techniques could lower energy usage without sacrificing real-time capability. Standardised benchmarking frameworks that incorporate both performance and energy thresholds would improve algorithm evaluation across diverse deployment scenarios.

For embedded computer vision designers, these findings demonstrate that energy efficiency must be evaluated alongside processing capability. Higher power consumption may be justified when it enables practical real-time functionality, whereas low power consumption without adequate performance limits the feasibility of deployment. TG-Lite shows that successful embedded solutions require balanced optimisation that treats energy and performance as connected, rather than competing, objectives.

ACKNOWLEDGMENT

This work was supported by the Ministry of Higher Education, Science and Technology of the Republic of Indonesia under Grant No. 123/C3/DT.05.00/PL/2025 (28 May 2025) and Contract No. 109/LL2/DT.05.00/PL/2025 (2 June 2025).

REFERENCES

Aharon, N., Orfaig, R., & Bobrovsky, B.-Z. (2022). *BoT-SORT: Robust associations multi-pedestrian tracking*. http://arxiv.org/abs/2206.14651

An, Y., Wu, J., Cui, Y., & Hu, H. (2023). Multi-object tracking based on a novel feature image with multi-modal information. *IEEE Transactions on Vehicular Technology, 72*(8), 9909–9921. https://doi.org/10.1109/TVT.2023.3259999

Ansari, M. A., & Singh, D. K. (2022). ESAR, an expert shoplifting activity recognition system. *Cybernetics and Information Technologies, 22*(1), 190–200. https://doi.org/10.2478/cait-2022-0012

Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). *Simple online and realtime tracking.* Proceedings - International Conference on Image Processing, ICIP, 2016-August, 3464–3468. https://doi.org/10.1109/ICIP.2016.7533003

Cao, J., Pang, J., Weng, X., Khirodkar, R., & Kitani, K. (2023). *Observation-centric SORT: Rethinking SORT for robust multi-object tracking.* Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2023-June, 9686–9696. https://doi.org/10.1109/CVPR52729.2023.00934

Du et al. (2023a). StrongSORT: Make DeepSORT Great Again. *IEEE Transactions on Multimedia, 25*, 8725–8737. https://doi.org/10.1109/TMM.2023.3240881

Ganesh, S. V., Wu, Y., Liu, G., Kompella, R., & Liu, L. (2023). *Fast and resource-efficient object tracking on edge devices: A measurement study*. https://arxiv.org/pdf/2309.02666

Global Road Safety | *Transportation Safety* | CDC. (n.d.). https://www.cdc.gov/transportation-safety/global/index.html

Guan et al. (2025). Multi-object tracking review: Retrospective and emerging trend. *Artificial Intelligence Review, 58*(8), 1–46. https://doi.org/10.1007/S10462-025-11212-Y/TABLES/8

Khalil, M. M., Saleh, S. N., Tawfik, N. S., & Elagamy, M. N. (2025). EF-StrongSORT: An enhanced feature StrongSORT model for multi-object tracking. *IEEE Access*, *13*, 53608–53620. https://doi.org/10.1109/ACCESS.2025.3554706

Li, X., Chen, C., Lou, Y., Abdallah, M., Kim, K. T., & Bagchi, S. (2024). *HopTrack: A real-time multi-object tracking system for embedded devices.* https://arxiv.org/pdf/2411.00608v1

Murat, A. A., & Kiran, M. S. (2025). A comprehensive review on YOLO versions for object detection. *Engineering Science and Technology, an International Journal, 70*, 102161. https://doi.org/10.1016/J.JESTCH.2025.102161

Nasir, R., Jalil, Z., Nasir, M., Alsubait, T., Ashraf, M., & Saleem, S. (2025). An enhanced framework for real-time dense crowd abnormal behavior detection using YOLOv8. *Artificial Intelligence Review, 58*(7), 1–50. https://doi.org/10.1007/S10462-025-11206-W/TABLES/15

OECD Urban Studies. (2023). *Smart city data governance.* OECD Publishing, OECD Urban Studies. https://doi.org/10.1787/E57CE301-EN

Precedence Research (2026). *Edge computing market size to hit USD 5,132.29 Bn by 2034*. https://www.precedenceresearch.com/edge-computing-market

Quasim, M. T., Khan, M. A., Algarni, F., & Alshahrani, M. M. (2021). *Fundamentals of smart cities.* Lecture Notes in Intelligent Transportation and Infrastructure, Part F1386, 3–16. https://doi.org/10.1007/978-3-030-60922-1_1

Rasheed, A. F., & Zarkoosh, M. (2025). Optimized YOLOv8 for multi-scale object detection. *Journal of Real-Time Image Processing, 22*(1), 1–12. https://doi.org/10.1007/S11554-024-01582-X/FIGURES/9

Sapkota et al. (2025). YOLO advances to its genesis: A decadal and comprehensive review of the You Only Look Once (YOLO) series. *Artificial Intelligence Review, 58*(9), 1–83. https://doi.org/10.1007/S10462-025-11253-3/TABLES/8

Satyajit Sinha. (2023, September 24). *Number of connected IoT devices growing 13% to 18.8 billion.* IoT Analytics. https://iot-analytics.com/number-connected-iot-devices/

Tsai, C. Y., & Su, Y. K. (2022). MobileNet-JDE: A lightweight multi-object tracking model for embedded systems. *Multimedia Tools and Applications, 81*(7), 9915–9937. https://doi.org/10.1007/S11042-022-12095-9/FIGURES/12

Varghese, R., & Sambath, M. (2024). *YOLOv8: A novel object detection algorithm with enhanced performance and robustness.* 2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems, ADICS 2024. https://doi.org/10.1109/ADICS58448.2024.10533619

Wojke, N., Bewley, A., & Paulus, D. (2017). *Simple online and realtime tracking with a deep association metric.* Proceedings - International Conference on Image Processing, ICIP, 2017-September, 3645–3649. https://doi.org/10.1109/ICIP.2017.8296962

World Health Organization. (2023, December 13). *Global status report on road safety 2023: Summary*. World Health Organization. https://www.who.int/publications/i/item/9789240086456

World Smart Cities Outlook 2024 | *UN-Habitat*. (n.d.). https://unhabitat.org/world-smart-cities-outlook-2024

Zhang et al. (2022). *ByteTrack: Multi-object tracking by associating every detection box.* Lecture Notes in Computer Science, 13682 LNCS, 1–21. https://doi.org/10.1007/978-3-031-20047-2_1

Zhang, Y., Wang, C., Wang, X., Zeng, W., & Liu, W. (2021). FairMOT: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision, 129*(11), 3069–3087. https://doi.org/10.1007/s11263-021-01513-4