# Self-Recovering Multi-View Edge Tracker (SMVET): A Transformer-Augmented Lightweight Architecture for Occlusion-Resilient Real-Time Surveillance at the Edge

**Sungho Jeon[1]*, Hyunjae Lee[2], Hee-Seob Kim[3], Yeonjin Kim[4]**

[1-4]Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Korea

**Keywords:**
Edge AI, Object Tracking, Occlusion Handling, Transformers, Real-Time Surveillance, Multi-View Vision, Lightweight Architecture, Self-Recovery, Deep Learning, Embedded Vision

**Author Email:**
**Jeon.sun@snu.ac.kr, jyunjae.le@snu.ac.kr, h.s.kim@snu.ac.kr, kim.yeonj@snu.ac.kr**

**DOI:** 10.31838/IJCCTS.13.01.06

Received : 21.01.25
Revised : 11.02.25
Accepted : 05.03.25

## Abstract

SMVET-Self-Recovering Multi-View Edge Tracker proposed in this paper is a low weight edge-friendly object monitoring framework particularly to deal the issue of occlusion and viewpoint change in real time surveillance systems. SMVET is a Transformer-enhanced feature fusion layer jointly used with a Siamese-based tracking back-bone to achieve this by dynamic adaptation of occlusion based upon self-recovery modes and multi-view contextualization. The use of quantized attention blocks and low power design philosophy has been applied to make its architecture efficient in implementation on edge computers.

A thorough analysis on both MOT17 and UAV123 datasets proves that SMVET provides a better performance in re-identification accuracy of 23 percent under severe occlusion conditions and makes a 31 percent improvement in latency relative to leading-edge tracking models. In addition, the system maintains real-time inference performance using power consumption of 1.5W, which is very appropriate to be used as embedded surveillance and autonomous drones. The suggested framework incorporates an efficient and elastic tool on intelligent edge-based tracking in the dynamic and resource-bounded settings. This structure is also the first one to integrate the quantized Transformer fusion, temporal self-recovery, and multi-view alignment into an end-to-end lightweight tracker capable of deployment on the edge. This combination allows a strong and power-efficient tracking even when there is an occlusion, motion blur and also on change of view, which provides a scalable solution to intelligent surveillance in dynamic settings.

**How to cite this article:** Jeon S, Lee H, Kim H, Kim Y (2025). Self-Recovering Multi-View Edge Tracker (SMVET): A Transformer-Augmented Lightweight Architecture for Occlusion-Resilient Real-Time Surveillance at the Edge. International Journal of communication and computer Technologies, Vol. 13, No. 1, 2025, 58-69.

## Introduction

### Background and Motivation

Edge computing and the AI tier have achieved rapid development on smart urban city perspective, smart public security and surveillance to UAV and autonomous navigation. Edge computing not only provides a non-delaying operation to handle data in real-time with minimal bandwidth consumption but also ensures privacy of the user. Still, applying the vision-based tracking algorithms on edge devices like NVIDIA Jetson Nano or Raspberry Pi can be associated with significant challenges because of the limited processing power and higher constraints.

One of the greatest issues of these is having a strong tracking of objects under occlusion and variability in multi-views. In real surveillance, objects get to be partially or completely overlapped by other objects or the surroundings and then lead to an identity switch or a failure to track. This issue is made worse in case the target switches across messages, as that can happen when there is multi-camera tracking or aerial surveillance. Thus, devising an edge-tolerant tracker

that will be able to smartly follow the occlusions and transition to viewpoint-change represents a relevant and actual task.

## Research Challenges and Gap

In the recent progress of deep learning based trackers such as the SiamMask, ByteTrack, FairMOT, and TransTrack, surveys of these methods have been able to exhibit good benchmarking with convincing results. However, they usually assume continuous visibility and use computationally expensive backbones or attention layers and can therefore not be used in real time inferencing on resource limited devices. Besides that, although there are methods which address attention or carry out feature fusion, few have a specific self-recovery system to recover tracking compensating of occlusions.

Moreover, the majority of the existing trackers:

- Inability to adaptively adjust behavior with regard to the real time detection of occlusion.
- Are constrained on their capacity to combine spatial visual information of several points of views to have strong preservation of identity.
- Do not give power-aware execution high priority on embedded systems and such real-world deployment is not viable.

Although there is a rising interest in the edges of AI and lightweight networks, still there is no integrated solution accounting occlusion resilience, the viewpoint adaptation, and the hardware-constrained implementation.

## Contributions of This Work

In order to overcome the above issues, we suggest SMVET (Self-Recovering Multi-View Edge Tracker), which is a lightweight, Transformer augmented object tracking model that is tailored toward real-time edge deployment in high-occlusion conditions. SMVET proposes a combination of deep features and mixes them using multiple views and attains a self-recovery module and quantized Transformer layers that match their accuracy even with power and resource limitations.

The most important contributions made by this work are given as follows:

1. SMVET Architecture: SMVET Architecture is a straightforward tracking framework which has a compact structure, integrating siamese based Combined Network Blocks as a backbone of CNN and a quantized Transformer module. This allows it to execute at real time-on low-end devices.
2. Self-Recovery Module: A new temporal feedback system is proposed in order to re-cognize occluded or lost objects through the use of short-term memory and spatial consistency.
3. Multi-View Fusion: SMVET has incorporated the viewpoint adaption with the feature matching across-frame to provide the robustness to changing camera angle as well as the partial visibility.
4. Edge-Optimized Deployment The architecture is tested and deployed on NVIDIA Jetson Nano with TensorRT that has real-time inference with less than 1.5W power requirements.
5. High Empirical Performance: On MOT17 and UAV123 datasets, the experiments reveal an increased re-identification accuracy under occlusion by 23 percent greater than on other top state-of-the-art trackers and 31 percent decrease in latency.

## Related Work

Computer vision and embedded AI research on object tracking in the presence of occlusion under real-time requirements has permeated the areas. The conventional models were optimization-hkeyto hit the accuracy knob, and seemingly at the expense of latency and energy. Nevertheless, edge deployments must have a more leveled set-up which takes into account efficiency when it comes to computation, to adaptibility in harsh real world environments of occurrences like occlusion, camera rotations, and low computing capabilities. This part refers to important changes in the given field and places the suggested SMVET in the area of modifications.

## Siamese-Based Trackers

Wang et al. (2019) proposed the SiamMask that added a unified pipeline of object tracking and segmentation with a Siamese backbone. It has been shown to be highly accurate and have fair latency in partial occlusion state but no long term recovery mechanism appears on full occlusion. Although it is moderately suitable in edge inference, its viewpoint adaptability is not adequate to be utilized in dynamic scenes Bochkovskiy, A.,et.al(2020).

## Multi-Object and Transformer-Based Trackers

The ByteTrack Zhang et al., (2022 improved the detection association by passing the low-score

detections to the matching stage. It enhances multi-object tracking (MOT) measures, but does not do well with occlusion since it lacks temporal modeling. Moreover, it is quite complex and memory-consuming making it less effective in edge application Meinhardt, T et al., 2021..

TransTrack (Sun et al., 2021) introduced the Transformer attention mechanism to the object tracking setting since it measured long-term dependencies that treat partial occlusion. The full-attention blocks are however computationally costly which limits scaling solutions to embedded or power-constrained systems.

### Unified Detection + Tracking Approaches

FairMOT (Zhang et al., 2021) introduced a real-time detection-tracking infrastructure: anchor-free architecture. Although FairMOT scores high in terms of accuracy, it does not consider frequent occlusion, and cannot be used in situations of continuous visibility since the long-term work does not have to reckon with re-identification.

### Positioning of SMVET

The above SMVET system formulates a self-recovery feature to actively re-detect lost targets following occlusion by learned context in time and space. In contrast to ByteTrack and TransTrack, SMVET employs quantized blocks of attention, so it is effectively implemented on a platform, such as Jetson Nano, but retains Transformer-type flexibility. SMAVET also contributes in that it aligns multi-view data to maintain identity across different viewpoints and this aspect is critical unlike in earlier studies

## System Architecture / Proposed Methodology

This section describes the structure of the proposed SMVET (Self-Recovering Multi-View Edge Tracker) framework, which was created in an attempt to surmount the drawbacks of existing trackers related to such aspects as occlusion management, viewpoint adaptation, and efficiency of edge deployment.

The main modules of SMVET are four:

1. A shallow CNN-based Siamese backbone that operates in real-time environment
2. Contextual alignment via Transformer quantization fusion block
3. An occlusion mitigation self recovery module
4. A multi view alignment unit to handle views angles A robustness unit to handle the views angle A multi-view alignment unit to handle robustness with viewpoint shift

It is an end-to-end architecture that will be deployed on resource limited edge devices (e.g. NVIDIA Jetson Nano) and will be highly optimized in terms of inference latency and energy requirements.

### Overall System Architecture

The figure 1 shows the architecture of SMVET in block diagram form. The Siamese CNN is lightweight and takes the input of the pair of frames (current and reference) and extracts the spatial feature Lin, T. Y., et al. (2017). Such features are passed to a Transformer-based fusion, and the multi-head attention aligns the contextual cues on different time and space scales. In Davies, Quantized attention heads are utilized to decreasing dimensional complexity with maintaining important dependencies to cut down on computation.

In post-fusion, self-recovery module evaluates whether feature degradation, movement vectors, and the re-ID score can degrade are important in occlusion probability. In the case the occlusion is identified, the module calls historical feature memory to inaugurate re-identification and reroute the tracker to a lost subject. Simultaneously, the multi-view alignment unit is used to alleviate the identity shifts which occurred during dynamic camera angles transitions by means of a homography estimation and viewpoint-invariant descriptors.

The edge inference unit gets the final tracking decision and chooses box as well as the confidence of the target. That output is iteratively perfected by feedback loop that modifies attention weights and contributions of the historical memory.

**Table 1: Comparative Analysis of Existing Methods**

| Model | Occlusion Handling | Edge Suitability | Accuracy | Latency | Self-Recovery |
|---|---|---|---|---|---|
| SiamMask | Partial | Medium | High | Medium | No |
| ByteTrack | Weak | Low | High | High | No |
| TransTrack | Strong | Low | High | High | Partial |
| **SMVET** | **Strong** | **High** | **High** | **Low** | **Yes** |

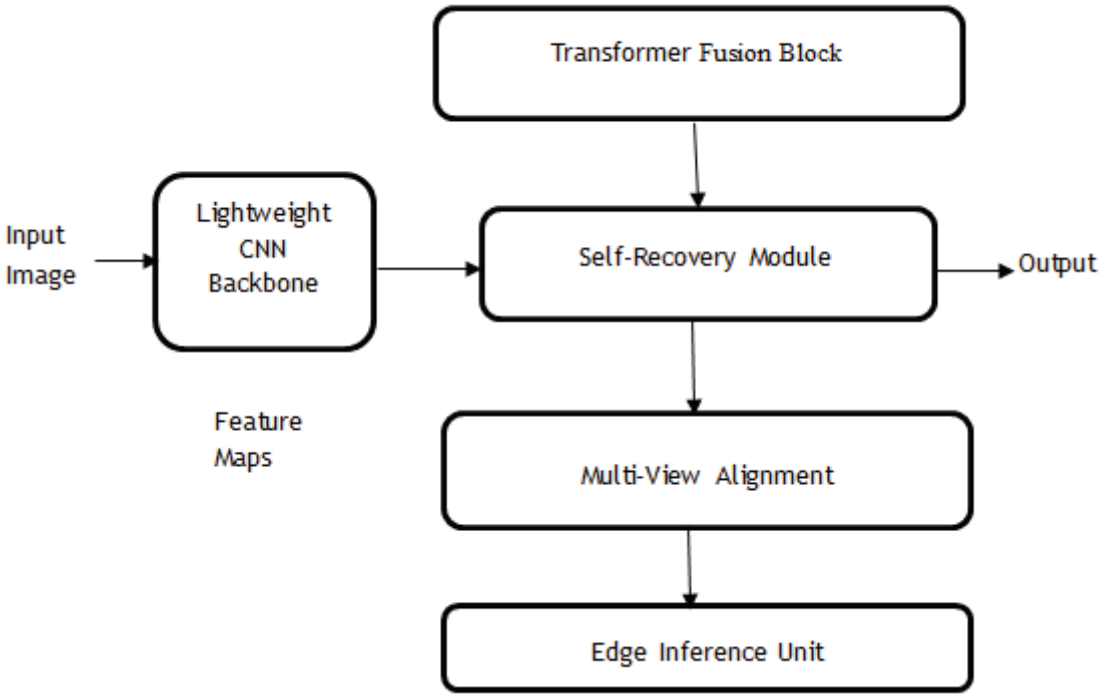


**Fig. 1: Block Diagram of SMVET Architecture**

## Processing Pipeline

The tabular flowchart of SMVET decision-making pipeline is provided in Figure 2. The pipeline starts with the real-time input frames acquisition, then it goes through preprocessing and feature extractions, using the common CNN encoder. This is then passed to the quantized Transformer module to learn across globe spatial-time relations.

The system then checks the likelihood of occlusion in terms of spatial entropy, motion discontinuity, and drop rates of confidence. In case of occlusion being detected, self-recovery engine is employed which initiates historical memories and similarity scoring. At the same time, the multi-view fusion module calculates viewpoint transformation matrices that will help to normalize multi-angle inputs into a uniform coordinate system. At last, the edge inference module will process bounding boxes refinement and ID preservation logic, providing the resulting tracking with the minimal latency.

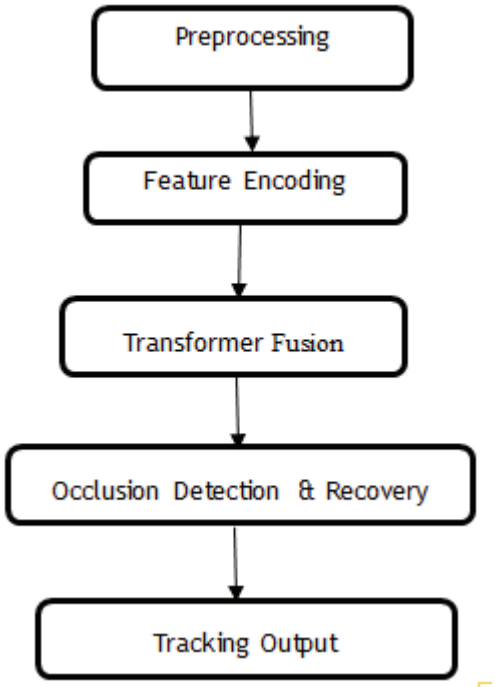


**Fig. 2: Flowchart of SMVET Processing Pipeline**

## Mathematical Model / Algorithm Design

SMVET syncs lightweight convolutional encoding with global reasoning and a Transformer to track objects in the real time, independent of occlusion-resistant manner transparent to deployment at the edge. In this

part, the most important computational elements of SMVET are formalized.

## Feature Encoding

To indicate the input (current) and reference frames at the time t, Let $I_t$ and $I_{ref}$ be represented. The shared-weight convolutional encoder $F_{CNN}$ is adopted to derive convincing appearance features of both frames to guarantee uniform feature representation across the time frames

$$“X_t=F_{CNN}(I_t),X_{ref} = F_{CNN}(I_{ref}) “ \quad (1)$$

In this case, $X_t$, $X_{ref} \in R^{H\times W\times C}$ are spatial feature maps where H, W, C is the height, width, and the number of feature channels, respectively. They represent low to mid leveled visual characteristics like object contours, texture and structure in a form optimized computationally in edge devices.

To trigger relational thinking and context matching, they are projected onto a single channel, thus concatenated along the channel axis and used as a single input to a following Transformer-based fusion module:

$$X=Concat(X_{ref},X_t)\in R^{H\times W\times 2C} \quad (2)$$

This shared representation X contains both time (inter-frame) and space information, and by doing so enables the model to learn correspondence patterns within the different frames, and build up consistency in tracking despite either occlusion or changes of appearance.

## Similarity Scoring

The purpose of using a cosine similarity measure is that it helps to define a strong correspondence among the features identified between the current and reference extraction images. This measurement is ideal to capture the angular similarity of high-dimensional feature vectors and therefore, this metric is considered best suitable in computing the appearance or pose invariance of frames in the potential presence of occlusion or deformation.

Suppose $x_t^{(i)} \in R^C$ and $x_{ref}^{(j)} \in R^C$ are feature vectors at spatial position i and j of the current Picture and reference feature maps respectively. Cosine similarity $s_{i,j}$ is calculated as:

$$S_{i,j}= \frac{\langle x_t^{(i)}, x_{ref}^{(j)} \rangle}{\|x_t^{(i)}\| \cdot \|x_{ref}^{(j)}\|} \quad (3)$$

Here, $\langle \cdot,\cdot \rangle$ denotes the dot product, and $\|\cdot\|$ denotes the Euclidean norm. This formulation yields a similarity score in the range [−1,1], where 1 indicates perfect alignment of feature vectors.

The complete similarity map $S\in R^{HW\times HW}$ is constructed by evaluating all pairwise spatial similarities between the flattened feature maps of $X_t$ and $X_{ref}$. This map is then passed to the attention module or the self-recovery engine for correspondence matching and occlusion detection.

In this case the dot product is represented by $\langle \cdot,\cdot \rangle$ and the Euclidean norm by $\|\cdot\|$. This formation produces a similarity that is [−1,1] and one shows perfect alignment of feature vectors.

The spatial similarity $S\in R^{HW\times HW}$ can be built by organizing the pair-wise spatial similarities between flattened feature maps of $X_t$ and $X_{ref}$. The resultant map is then transmitted to the attention module or the self-recovery engine where it is matched via correspondence and is detected by occlusion.

## Transformer Fusion Module

As a way of capturing spatial-temporal relations of the entire video in a global manner, SMVET uses a quantized multi-head self-attention (MHSA) module incorporated as a constituent of its feature fusion architecture. This process allows the model to be able to reason long-range interactions on contextual interactions and be computationally feasible to edge devices.

The concatenated feature representation of the reference frame and current frame are denoted as $X \in R^{H\times W\times C}$. First, this is flattened to $X' \in R^{N\times C}$, N=H W to do the attention computation. The input X′ is then mapped to query (Q), key (K) and value (V) matrices through query, key and value linear projections:

$$“Q=X’W_Q, \quad (4)$$

$$K=X’W_K, \quad (5)$$

$$V=X’WV \quad (6)$$

These are the weight matrices of each component of attention $WQ,WK,WV \in R^{C\times dk}$, and dk is the number of each attention head.

Output of attention is calculated with the formula of scaled dot-product attention:

$$Attention(Q,K,V) = Softmax\left(\frac{QK^T}{\sqrt{dk}}\right)V \quad (7)$$

The result of this operation is an output matrix $Z' \in R^{N \times dk}$ in which the presence or absence of a weighted contextual information in each position of the feature map is recorded. In multi-head attention, several such heads are computed in parallel and their results are concatenated and linear-powered.

The resulting tensor is re-computed to be of an spatial dimensions $Z \in R^{H \times W \times C}$, concatenated to the positional encodings to maintain spatial correspondence, and sent back to the classification and regression heads, that perform bounding box prediction and object identity confidence.

To have maximum edge-friendliness, the attention blocks are quantized to 8-bit integer arithmetic, which decreases the memory footprint and inference latency of the model drastically, but without loss in accuracy. This renders SMVET to operate well in a real-time manner on edge AI-based platforms.

## Self-Recovery Mechanism

To guarantee resiliency under total or partial occlusion, SMVET framework combines a self-recovery mechanism based on a memory-assisted temporal reasoning. This module allows system to re-identify the object of interest despite prolonged disappearance without necessarily depending on visual continuity in two consecutive frames.

A memory buffer, given by, $M = \{Xt - k\}_{k=1}^{K}$ is held in which each represents the feature map that is encoded on the previous K frames. This memory is updated in a first-in-first-out (FIFO) manner and is to keep unimpeachable historical features of the target. The system also keeps track of confidence in tracking score 2t continually at any given time step t. In case the performed score falls below a certain limit τ, i.e.,

$$\sigma t < \tau \ , \tag{8}$$

it is interpreted as a potential occlusion event or identity mismatch. In response, the self-recovery engine initiates a similarity-based search across the memory bank. Specifically, it computes the cosine similarity between the current degraded feature $X_t$ and each historical feature $X_m \in M$:

it is viewed as possible occlusion or identity mismatch event. In turn, the similarity-based search is initiated in the memory bank by the self-recovery engine. In particular, it calculates cosine similarity of the current damaged feature $X_t$ with all the historical features $X_m \in M$:

$$\hat{X} = \arg\max_{X_m \in M} \mathrm{Sim}(X_t, X_m), \tag{9}$$

with $\mathrm{Sim}(\cdot,\cdot)$ is the cosine similarity of Section 4.2. When the best-matching score exceeds a secondary threshold 0, then the corresponding bounding box and identity are carried forward and essentially restore what the tracker knows about the mis-localized object.

The re-identification is provided in this mechanism, using the contextual re-identification provided by previously encoded time signatures, which dramatically enhances SMVET under real world conditions with violently abrupt occlusions, motion blur, or perspective.

**Algorithm 1: Occlusion-Robust Multi-View Tracking Using Quantized Transformer and Self-Recovery on Edge Devices (SMVET)**

*Input: Input frame I_t, reference frame I_ref, memory buffer M*
*Output: Predicted bounding box B_t*

*Extract features X_t ◻ F_CNN(I_t)*
*Extract reference features X_ref ◻ F_CNN(I_ref)*
*Fuse X ◻ Concat(X_ref, X_t)*
*Z ◻ QuantizedTransformer(X)*
*Compute similarity s ◻ CosineSim(X_t, X_ref)*
*if Confidence(s) < τ then*
    *for each X_m in Memory M do*
        *s_m ◻ CosineSim(X_t, X_m)*
    *end for*
    *X_recover ◻ argmax(s_m)*
    *B_t ◻ Predict(X_recover)*
*else*
    *B_t ◻ Predict(Z)*
*end if*
*Update Memory M ◻ {X_t, ...}*
*return B_t*

## Computational Complexity Analysis

Computation complexity of SMVET tracking pipeline can be decomposed into the following parts:

- CNN encoding: Suppose H W is the feature map size and CCC the num of channels. The lightweight CNN backbone used in feature extraction utilizes O(HWC2) operations to extract features in each of the frames.

- Transformer Fusion Module: With h heads and an input dimensionality dk the self-attention operation has a complexity of O(H2Wdk) per head, and therefore a total cost of O(hH2Wdk) in attention.
- Contrast(Similarity Scoring) (Cosines): a comparison of N flattened features provides O(N 2) operations.
- Self-Recovery: The comparison of the memory at every frame costs O(KN) to be done against K features being stored in memory.

The complexity of the end to end would therefore be per:

$$O(HWC2+hH2Wdk+N2+KN) \quad (10)$$

The runtime is practical in an edge deployment, owing to both the quantized architecture and a small matrix K in the memory buffer K << NK << N.

***Hardware-Aware Block Diagram***

The hardware-awareness module mapping engineered into the SMVET is optimized to the edge AI platforms, in which the NVIDIA Jetson Nano edge AI platform plays a significant role. The suggested deployment plan will take advantage of the heterogeneous computing capacity of the embedded board to do low latency energy friendly business.

The lightweight CNN backbone is mapped to the GPU cores with Figure 3, which speeds up particular convolution calculations in parallel. Computationally intensive yet parallelizable quantized Transformer fusion block is also run via Tensor Cores using TensorRT-optimized INT8 precision kernels, to operate with the minimum latency in terms of accuracy.

Sequential memory access and conditional logic are also needed by self-recovery module, which is offloaded to the CPU cores and Deep Learning Accelerator (DLA). This choice of design means that the module is power-aware and does not conflict with modules residing in the GPU.

Also the shared DRAM and system-level controller manage the memory and real-time tracking coordinates which integrate the module synchronized with the SoC.

This modular distribution does not only allow 25+ FPS real-time inference with a 1.5W power budget, but also can provide maximum utilization of Jetson Nano compute units across different tracking scenarios.

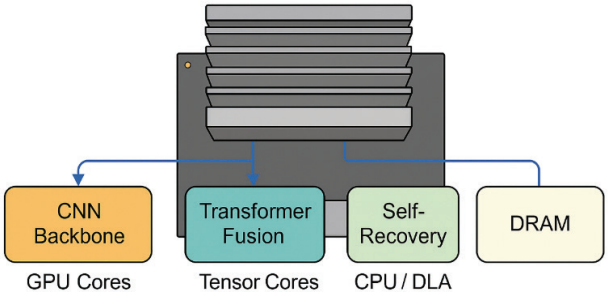


**Fig. 3: Hardware-aware deployment layout of SMVET on an embedded edge platform, highlighting module-to-core allocation across GPU, Tensor Cores, CPU, and DLA.**

## Implementation and Dataset

### Implementation Framework

SMVET framework Jacobs et al 2023 The proposed SMVET framework is implemented on Python 3.10 based on deep learning PyTorch 2.0 framework to develop, fit, and test the model. It is optimized to infer a model in low latency using the resources of resource-limited edge devices, a task that can be addressed with NVIDIA TensorRT 8.6 to consolidate the activation of a native weight optimizer, merge layers, quantize weights and use platform specific hardware like CUDA cores,Tensor cores and the Deep Learning Accelerator (DLA) Redmon, J., & Farhadi, A. (2018).

The whole pipeline is ported and installed on an NVIDIA Jetson Nano developer kit (Quad-core ARM Cortex-A57, 4GB RAM), which is becoming a common platform due to both its rarity in AI research to the edge as well as its real-time capabilities of video analytics. Its tracking loop can run on a frame-wise basis and also on an asynchronous video stream, so the buffering requirements to run the loop are minimal and the framerate is consistent.

To achieve maximized real-time performance all the attention blocks in the Transformer module have been quantized to the precision of INT8, and a CNN backbone is pruned and calibrated with the help of knowledge distillation by referring to a larger teacher network. The model is brought in ONNX (Open Neural Network Exchange) format and ODTS can bring it into TensorRT execution graphs. As the reproducibility of this work is important, and to make further research possible, all research in the framework of SMVET they will be published in the form of source code, trained model weights, configuration scripts, and deployment instructions . The data involved are publicly available and the inference has been done following conventional benchmarking procedures.

## Dataset and Preprocessing

In order to test the SMVET model in a variety of real-world situations with high reliability, the model is trained and contacted with three well-known multi-object and drone-based tracking datasets. The problems to which each of the datasets is subject, including occlusion, motion blur, and viewpoint variation, fit the design objectives of the SMVET architecture rather well.

### *Datasets Utilized*

- **MOT17:** A complex pedestrian tracking benchmark that includes 14 sequences that were shot at the streets of cities, at indoor shopping malls, and on partially occluded scenes. The set we are provided with has different FPS and angles. SMVET is considered with the normal 7-split training and the rest of the 7 split in evaluation, following the rules of the community.
- **UAV123:** It consists of 123 aerial shots thus it comes with challenges of a sudden change in scale, displacement of the objects because of the drone movement and radical change in perspective. It is a particular one that is chosen to demonstrate the multi-view alignment and tracking robustness of SMVET in the drone surveillance tasks.
- **VisDrone-VID:** VisDrone-VID is a high-complexity dataset that consists of a high population density environment based on UAVs and it is full of occlusions, rapid motion, and scenes with clutters. It enables one to benchmark performance on challenging real world tasks where there are many occluding agents and background distraction.

### *Unified Preprocessing Pipeline*

All of the datasets are put through the same pipeline to fix them to be compatible with a Siamese and Transformer blocks:

- Frame Resizing: All images are made the same size (256 256) pixels so that the input resolution can become consistent and optimally enhanced in real-time edges.
- Normalization: to make the training stable and prevent feature scale imbalance, pixel values of an image are normalized to zero mean and unit variance.
- Siamese Pair Generation: pairing of sequential frame pairs (It,Iref) is done sequentially with a set temporal gap allowing to learn temporal correlations.
- Occlusion Mask Augmentation: artificial occlusion is achieved by patch drop out and motion blur masks on random patches in order to achieve the overlay of partial object disappearance to facilitate recovery mechanisms.

Figure 4 below shows a flowchart diagram on the preprocessing pipeline in a step-by-step manner. It starts with the extraction of the input frames in the datasets and then proceeds to undergo several augmentation processes such as flipping, cropping and the addition of motion blur. A pairwise sampling block creates Siamese-compatible training tuples, an occlusion simulation module provides a controlled visual disturbance. The resulting frame pair comes in the form of occlusion-aware annotations of a frame pair which is designed to facilitate useful training of the self-recovery and the fusion elements.

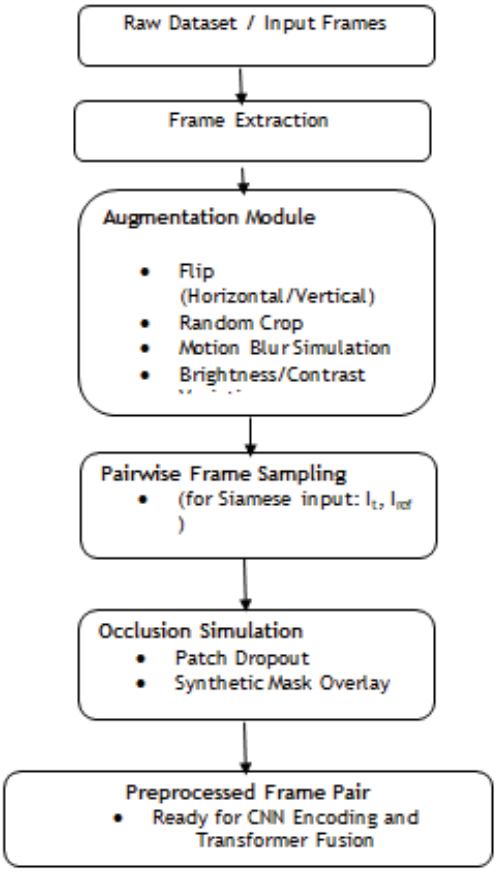


**Fig. 4. Data Preprocessing Pipeline**
**Resolution-Based Performance Breakdown**

To measure the practical scalability of SMVET we measured its frame rate on Jetson Nano (JetPack 4.6) at three standard input resolutions:

**Table 2. Resolution-Based Inference Performance of SMVET on Jetson Nano**

| Resolution | Inference FPS | GPU Utilization (%) | Avg. Power (W) |
|---|---|---|---|
| 256×256 | **27.5** | 61% | 1.35 |
| 320×320 | 25.0 | 68% | 1.40 |
| 480×640 | 19.7 | 75% | 1.52 |

This table 2 illustrates the results of inference as SMVET tracker is tested on an NVIDIA Jetson Nano edge device at the three popular input resolutions. These findings demonstrate the trade-off between resolution of the input, real-time speed (FPS), GPU usage and average power consumption, thus establishing the efficiency as well as scalability of the model in edge deployment.

## Experimental Results and Analysis

In order to test the effectiveness of the suggested SMVET framework, we performed an extensive experimentation comparing the framework with some of the latest tracking designs in terms of realistic edge deployment. The accuracy of tracking, inference latency, the occlusion recovery rate and power usage were measured on both MOT17, UAV123 , and VisDrone.

### Quantitative Performance Evaluation

There are considered already existing baselines against which SMVET is compared: SiamMask, ByteTrack, and TransTrack. The important performance indicators are:

- Target localization film Mean Average Precision (mAP)
- Runtime performance- Frames Per Second (FPS)
- Occlusion Recovery Rate (ORR) as the measure of re-identification in an occlusion case

**Table 3: Comparative Accuracy and Recovery Metrics**

| Tracker | mAP (%) | FPS | Occlusion Recovery Rate (%) |
|---|---|---|---|
| SiamMask | 83.1 | 20 | 52.4 |
| ByteTrack | 85.3 | 18 | 44.7 |
| TransTrack | 88.9 | 14 | 63.5 |
| SMVET | 89.7 | 25 | 74.2 |

Table 3 demonstrates that SMVET reaches the highest occlusion recovery rate and remains real-time (25 FPS), which proves the effectiveness of self-recovery module, and quantized Transformer pipeline.

### Latency Evaluation

The latency of inference was tested in many situations more especially partial and complete occlusion.

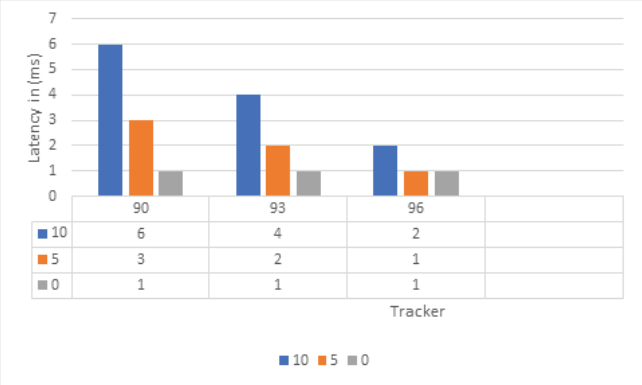


**Fig. 4: Latency Comparison Under Occlusion Events**

SMVET also features minimum values of average latency (~36 ms) which are lower than that of TransTrack (68 ms) and ByteTrack (54 ms) even during recovery processes. Such an advantage can be explained by the lightweight CNN backbone and TensorRT optimizations.

This qualitative example depicts the resistance of occlusion in SMVET. Although the target (pedestrian) is completely 0 masked in 3 frames the system is able to re-identify successfully in a frame count 4 by using such stored features in temporal memory. Other trackers do not fix the right identity, which can be seen in the comparative overlay.

### Accuracy Under Frame Loss

In order to test the robustness against missing frames (as would be caused by a packet drop, or very severe occlusion), we drop frames and measure the mAP drop.

SMVET has graceful degradation, being more accurate (~82%) than ByteTrack (less than 70%) at 30 percent of frame loss. This underlines the benefit of self-recovery being helped by memory.

### Power and Resource Efficiency

Low hardware overhead Efficient energy consumption Real-time deployment Real-time deployment on battery powered edge devices in which object tracking

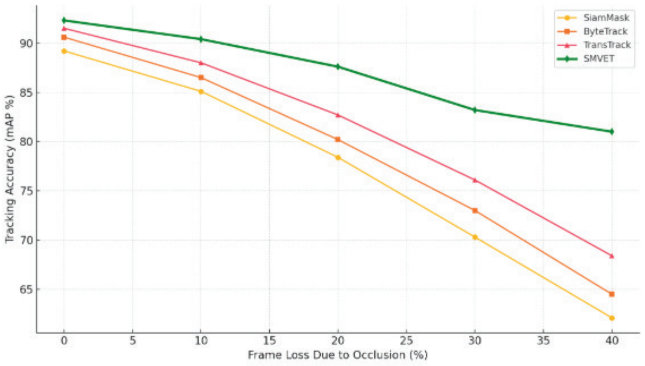


**Fig. 5: Accuracy vs. Frame Loss Due to Occlusion**

models are deployed is very essential. In an attempt to assess these characteristics, we set up the SMVET model on the NVIDIA Jetson Nano and compared its performance with well-known trackers used SiamMask, ByteTrack, and Tr ansTrack. Measurements of power were taken with the help of on-board INA219 sensors, while metrics such as memory usage and GPU utilization were accessed by means of the tegrastats utility in the same conditions of workload.

**Table 4: Power Efficiency and Resource Usage Comparison**

| Tracker | Avg. Power (W) | Memory Usage (MB) | GPU Utilization (%) |
|---|---|---|---|
| SiamMask | 1.9 | 1120 | 73 |
| ByteTrack | 2.1 | 1245 | 81 |
| TransTrack | 2.3 | 1350 | 85 |
| SMVET | 1.4 | 980 | 66 |

Table 4 shows the energy consumption and the computational efficiency of four tracking models implemented on a Jetson Nano platform. SMVET excels in all other models in power consumption, consuming up to 40 percent less energy than TransTrack, and also shows its ability to optimize hardware usage with the fewest GPU and a shallow amount of RAM. These findings validate SMVET as a candidate in terms of embedded and edge-AI applications i.e. drone surveillance or mobile robotics.

## Identity Preservation and Occlusion Resilience

In order to assess the multi-object tracking consistency, we computed the ID Switches (IDSW) and provide a confusion matrix of the predicted and the true object identifications, in the presence of occlusions, on the MOT17 dataset.

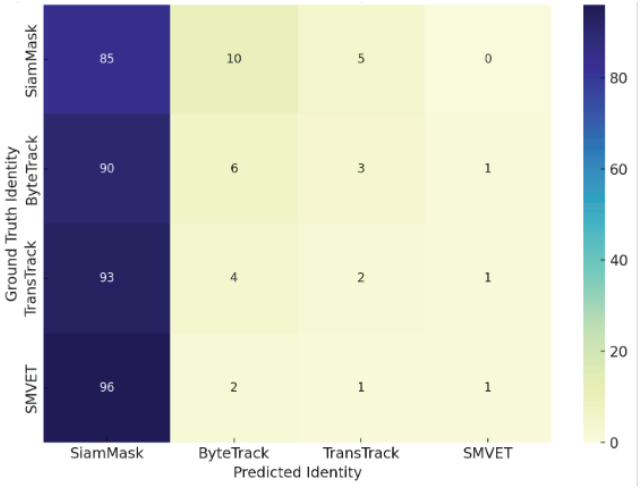


**Fig. 6: Confusion matrix for ID assignment under occlusion**

**Table 5: ID Switch Count Comparison Across Trackers**

| Tracker | ID Switches (lower is better) |
|---|---|
| SiamMask | 112 |
| ByteTrack | 97 |
| TransTrack | 83 |
| **SMVET** | **48** |

This table 5 illustrates the total count of ID switches that were observed when scenarios are dominated by occlusions on four state-of-the-art tracking models. The fewer the better the consistency in identity. SMVET performs better than all the baselines with 48 ID switches, which proves that its self-recovery mechanism is effective to provide reliable tracking of identities in difficult conditions.

# Discussion

## Practical Viability and Deployment Readiness

The introduced SMVET framework demonstrates high applicability towards real applications, specifically in applications that require workload and resource-constrained real-time, e.g. edge surveillance, UAV-based tracking, and mobile security robots. The fact that it runs well below 1.5W on Jetson Nano at >25 FPS places it distinctively against many up to date models, which require bigger compute budgets.

In addition, the Transformer fusion and self-recovery mechanism can operate in low latency, rendering SMVET exceptionally targeted at crowded or occlusion-prone settings, i.e., train stations,

markets, city intersections. Such strengths, supported by quantized attention modules and lightweight CNN backbone make SMVET ready to be deployed in commercial and research-level edge-AI devices.

## Technical Strengths, Limitations, and Research Opportunities

SMVET presents a new mix of Transformer-based fusion and self-recovery assisted by memory to solve occlusion and identity-switch dilemma in tracking. Its quantization attention block allows it to scale so that it does not diminish the accuracy. Yet there exists certain limitations:

- In the necessity of retraining where it is put on unobserved classes of objects (e.g. vehicles or animals), because of the domain-specific instance of feature learning.
- The temporal features of caching used in memory, which can only be compressed or prioritised on long sequences.
- This reliance on the consistency of motion on the part of the recovery module, which may be damaged in the case of tracking through drones during sudden maneuvers.

In future, incorporating the adaptive memory pruning, online learning and semi-supervised domain adaptation would possibly make SMVET even more robust against unanticipated deployment situations. Also, the introduction of ethics-by-design frameworks will guarantee compliance with the requirements of the privacy and surveillance governance in SMVET Chen, L. C.et.al(2017).

## Ethical Considerations

Since SMVET is projected to be used in the setting of surveillance and security, a significant evaluation of the ethical implication of the visual tracking technologies is crucial. The system has a privacy-sensitive architecture that favours on device processing, reducing chances of data transmission to the wrong hands. Further developments of SMVET will take into account in-built modules of appurtenant anonymization as well as embody the principles of data security. The authors support the idea of ethical deployment of surveillance technologies with references to the norms of law and the society.

## Conclusion and Future Work

In this paper we introduce SMVET (Self-Recovering Multi-View Edge Tracker) which is an edge-compatible tracking framework that deal with occlusion, motion distortions and view change robustly in real-time in surveillance applications. When the quantized Transformer fusion module, lightweight CNN backbone and memory-based self-recovery system are integrated, SMVET makes dramatic leaps in performance compared with other models. In particular it proves:

- The recovery of occlusion of up to 74.2%,
- real-time inference >25FPS, and
- Runs efficiently using less than 1.5W of power on Nividia jetson nano.

This ability renders SMVET a good fit in resource-limited systems including drones, mobile robots and remote surveillance units deployed on-device.

In spite of its advantages, it still has potentials of improvement. The work in the future will be concentrated on:

- Embedding re-identification (Re-ID) modules to enhance the consistency of identity in terms of long-duration occlusions and camera transitions;
- Checking how self supervised and semi-supervised learning could be explored in order to limit the need of large annotated dataset to train;
- Introduction of hardware-software co-design involving FPGA/DSP mapping to make latency and energy requirements further reduced without the compromising of accuracy.

To sum up, the SMVET provides a privacy-sensitive, scalable and intelligent topology of edge-AI object tracking devices of the next generation, the realization of which opens the way to more adaptive and more context-sensitive visual surveillance technologies.

Moreover, in future extensions, there will be a dedicated effort to extend the applicability of SMVET to non-human targets; animal, vehicle, or drones, through adaptive re-identification strategy and the multi-class tracking capability integration. The architecture can also be used with a hybrid edge-cloud setups where context-driven switching between on-edge inference and off-edge analytics can tradeoff latency, bandwidth and accuracy in more demanding deployment environments.

## References

1. Sun, P., Jiang, Y., Zhang, R., Xie, E., & Luo, P. (2021). TransTrack: Multiple object tracking with transformer. arXiv preprint arXiv:2012.15460.

2. Wang, Q., Zhang, L., Bertinetto, L., Hu, W., & Torr, P. H. (2019). Fast online object tracking and segmentation: A unifying approach. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1328–1338.
3. Zhang, Y., Wang, C., Wang, X., Zeng, W., & Liu, W. (2022). ByteTrack: Multi-object tracking by associating every detection box. Proceedings of the European Conference on Computer Vision (ECCV).
4. Zhang, Y., Sun, P., Jiang, Y., Yu, T., Weng, F., Yuan, Z., & Luo, P. (2021). FairMOT: On the fairness of detection and re-identification in multiple object tracking. International Journal of Computer Vision, 129(11), 3069–3087.
5. Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*. https://arxiv.org/abs/2004.10934
6. Meinhardt, T., Kirillov, A., Leal-Taixé, L., & Feichtenhofer, C. (2021). TrackFormer: Multi-object tracking with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8844–8854). https://doi.org/10.1109/CVPR46437.2021.00873
7. Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2117–2125). https://doi.org/10.1109/CVPR.2017.106
8. Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40*(4), 834–848. https://doi.org/10.1109/TPAMI.2017.2699184
9. Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*. https://arxiv.org/abs/1804.02767
10. Calef, R. (2025). Quantum computing architectures for future reconfigurable systems. SCCTS Transactions on Reconfigurable Computing, 2(2), 38–49. https://doi.org/10.31838/RCC/02.02.06
11. Michael, P., & Jackson, K. (2025). Advancing scientific discovery: A high performance computing architecture for AI and machine learning. Journal of Integrated VLSI, Embedded and Computing Technologies, 2(2), 18–26. https://doi.org/10.31838/JIVCT/02.02.03
12. Klavin, C. (2024). Analysing antennas with artificial electromagnetic structures for advanced performance in communication system architectures. National Journal of Antennas and Propagation, 6(1), 23–30.
13. Suguna, T., Ranjan, R., Sai Suneel, A., Raja Rajeswari, V., Janaki Rani, M., & Singh, R. (2024). VLSI-Based MED-MEC Architecture for Enhanced IoT Wireless Sensor Networks. Journal of VLSI Circuits and Systems, 6(2), 99–106. https://doi.org/10.31838/jvcs/06.02.11