ADARE-HD: Adaptive-Resolution Framework for Efficient Object Detection and Tracking via HD-Computing

Mohamed Mejri, Chandramouli Amarnath and Abhijit Chatterjee
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, Georgia 30332–0250

***Abstract*—Efficient and low-energy camera signal processing is critical for battery-supported sensing and surveillance applications. In this research, we develop a video object detection and tracking framework which adaptively down-samples frame pixels to minimize computation and memory costs, and thereby the energy consumed, while maintaining a high level of accuracy. Instead of always operating with the highest sensor pixel resolution (compute-intensive), video frame (pixel) content is down-sampled spatially, to adapt to changing camera environments (size of object tracked, peak-signal-to-noise-ratio (i.e, PSNR) of video frames). Object detection and tracking is supported by a novel video resolution-aware adaptive hyperdimensional computing framework. This leverages a low memory overhead non-linear hypervector encoding scheme specifically tailored for handling multiple degrees of resolution. Previous classification decisions of a moving object based on its tracking label are used to improve tracking robustness. Energy savings of up to 1.6 orders of magnitude and up to an order of magnitude compute speedup is obtained on a range of experiments performed on benchmark systems.**

***Index Terms*—hyperdimensional computing, Resolution Adaptation, object detection and tracking,**

I. Introduction

The continuous increase in available camera resolution over recent years has led to greater computational complexity and overhead in real-time object classification for multi-object tracking, especially for low-power edge applications [1]. A significant body of work presupposes fixed image resolution, thus accruing significant overhead when using modern high-resolution camera systems [2] across diverse video environments. The core idea of this research is based on the notion that significant reductions in power consumption can be obtained by modulating the amount of computation performed in proportion to the quality of the video being processed at any time. As an example, for object detection and tracking (labeling), the pixel resolution allocated to each tracked object may vary with the degree of lighting, frame image quality as induced by fog or snow, or distance from the camera to save power (lower resolution for higher quality video and vice versa). There are two aspects to this approach: (a) the need for a control policy to determine under what conditions to switch the pixel resolution of a tracked object from lower (achieved by pixel downsampling) to higher or from higher resolution to lower, across video frames and (b) efficient processing of down-sampled frame images; here we leverage a novel image encoding scheme enabled by hyperdimensional computing (HDC) [3]. Note that different parts of a frame image may be down-sampled differently and the resulting variable-resolution video is processed on a frame-to-frame basis in an adaptive manner.

The key contributions of this paper are as follows:

- ADARE-HD uses a novel approach for video object detection and tracking that *combines adaptive pixel downsampling* as a means of adaptive resolution video processing *with hyperdimensional computing* for minimal energy consumption. Adaptive resolution video (frame/image) processing is performed within each frame (spatial dimension) through a novel policy that ensures efficient object classification as well as object recognition (i.e, feature matching) at optimal resolution to balance overhead and accuracy.
- A novel, low memory overhead non-linear hyperdimensional vector encoding scheme specifically designed for adaptive HDC systems is developed. When combined with the resolution adaptation module and a detection tracking algorithm, ADARE-HD is *6.23* times faster than deep learning based object detection and tracking [4] and *1.6* more energy efficient.
- In uncertain video environments, mislabeling of objects can occur due to low image (frame) fidelity. A novel approach is developed that takes into account previous classification decisions of a moving object based on its tracking label for generating reliable classification decisions.

II. Prior Work

Prior work on adaptive resolution algorithms for object detection and tracking is based on signal processing techniques such as background subtraction, quadtree segmentation, and histogram of oriented gradients (HOG) feature extractors [5]–[7]. To enhance their accuracy, a trend has been to augment such solutions with algorithms relying on deep learning. However, these are compute-intensive and power-hungry. Lightweight object detection algorithms [8] are commonly employed for multi-object tracking tasks in conjunction with modern trackers [9], [10] and feature matching networks

Funding agency identification removed for review.

[11]. Although these techniques provide high accuracy driven by deep learning, their high memory access overhead is not conducive for integration with edge devices. Dynamic adjustment of image resolution has been explored as a means of reducing the computation overhead of deep learning systems [12]. However, memory overhead and energy use remains a problem for deep learning based computer vision [13].

The problem of low-overhead, intelligent adaptive-resolution algorithm design is addressed in this work by building on the unique capabilities of hyperdimensional computing (HDC) [3]. We introduce ADARE-HD, an innovative adaptive-resolution framework for object classification in video sequences. ADARE-HD integrates a prior lightweight motion-based object detection mechanism with a novel multi-resolution tracking and hyperdimensional object classification algorithm. There has been research in the past on dynamic resolution networks [12] that adapt the resolution of the camera image using a neural network. Concurrently, bio-inspired hyperdimensional computing systems have been proposed for object recognition [14]. Such systems have diverse applications ranging from voice recognition [14] to bio-informatics [14]

Several encoding systems have been designed for hyperdimensional computing. These include CNN-based encoding [15], kernel-based encoding [16], and binary encoding [14]. These represent different tradeoffs between complexity and accuracy of the underlying learning algorithms. Adaptive algorithms using HDC were introduced in [17]. Two different encoding systems associated with high and low dimension class hypervectors respectively, are used. An HDC-based decision algorithm forwards the input signal to the appropriate classifier. There has also been significant prior work on object tracking and detection [4], [18], [19]. These combine deep learning with classical computer vision techniques such as background subtraction [20] for object detection. The associated algorithms have been used for single camera multi-object-tracking tasks such as the MOT [21] challenge, or a combination of multi-camera and single camera multi-object tracking tasks such as the AI CITY Challenge 2022 Track 1 [22]. While there has been independent research on adaptive resolution object detection and tracking and in the domain of hyperdimensional computing for vision applications, this research *combines adaptive resolution video processing with hyperdimensional computing for energy-efficient, accurate multi-object tracking and classification.* In the following, an overview of the proposed approach is first presented, followed by algorithmic details and experimental results.

III. Overview

Fig. 1 gives an overview of the operation of ADARE-HD. The goal is to track objects in video sequences and classify them (such as trucks vs cars on roadways) with minimal amounts of computation. This is described below.

Step-1: Motion-based Object Detection: In Block-1 of Fig. 2, moving objects in the input video are detected without the high overhead of a deep learning-based approach. The detection of moving objects is performed after background subtraction (by taking an absolute difference between successive frames), non-maxima suppression and dilatation to suppress small objects and objects with a high overlap

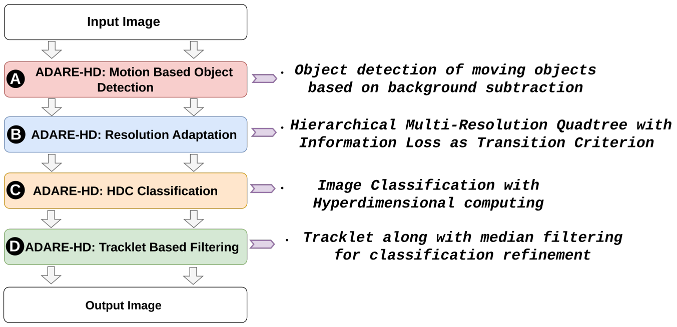
Input Image
A ADARE-HD: Motion Based Object Detection
Object detection of moving objects based on background subtraction
B ADARE-HD: Resolution Adaptation
Hierarchical Multi-Resolution Quadtree with Information Loss as Transition Criterion
C ADARE-HD: HDC Classification
Image Classification with Hyperdimensional computing
D ADARE-HD: Tracklet Based Filtering
Tracklet along with median filtering for classification refinement
Output Image

Fig. 1: ADARE-HD: Overview

Step-2: Resolution Adaptation: Our approach is grounded in the understanding that object recognition does not require high-resolution imagery. In this step (Block-2 of Fig. 2), the system selects the *minimum resolution that allows objects to be classified accurately* in video sequences. Image frames are represented using quadtrees with lower levels of the quadtree representing increasing image resolution. The quadtree depth is modulated dynamically across different regions of the image based on information loss as measured by PSNR. The optimal resolution is one that achieves a low information loss score above a specified threshold.

Step-3: HDC Classification: In contrast to deep learning-based classification techniques, we adopt a hyperdimensional computing framework [3] This module consists of a set of N hyperdimensional computing based classifiers, where N refers to the number of resolution levels. Each classifier has a lightweight convolutional Radial Basis Function (RBF) [23] kernel based encoding. An independently trained classifier is attached to each level of pixel resolution (obtained from Step-2).

Step-4: Tracklet-based Filtering: We also address the challenge of consistent object classification when the same object is viewed from different angles and positions. By leveraging the consistency of object classes across frames, this step refines the classification results of Step-2. The classification history (also called the *tracklet*) of each object is populated with its predicted label. A median filter is then applied to each tracklet, smoothing out the classification label time-series to ensure consistency.

IV. Methodology and algorithms

A. ADARE-HD: Motion based Object Detection

Object detection algorithms range from simpler approaches such as Cascade Haar [24] to more complex methods based on deep learning-based detectors (e.g, YOLOV7 [25]) . Although these detectors vary in computational overhead and classification accuracy, most of them require extensive training. In this paper, our objective is to effectively classify and track moving objects in scenes captured by a fixed camera. We opt for an unsupervised, motion-based method for object detection derived from state-of-the-art [26]. The detection pipeline comprises: 1- background subtraction (i.e., $D_t = \|I_{t+1} - I_t\|$, where I_{t+1} and I_t denote frames at time t+1 and time t, respectively), This pipeline is followed by a 2- contours detection and a non-maxima suppression post-processing. Figure 2 shows the object detection algorithm step by step.

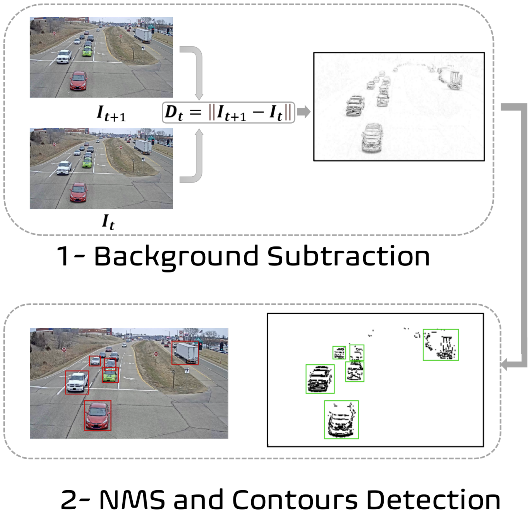
I_{t+1}
$D_t = \|I_{t+1} - I_t\|$
I_t
1- Background Subtraction
2- NMS and Contours Detection

Fig. 2: Object detection steps

B. ADARE-HD: Resolution Adaptation

In this section, we present the Adaptive Resolution Hierarchical Deep Convolutional (ADARE-HD) mechanism , which aims to make predictions based on the optimal resolution level for a given video frame object, balancing overhead and accuracy. The aim of this strategy is to identify the lowest resolution at which the object label can be accurately predicted. The fundamental concept here is that the quality of information diminishes progressively with each reduction in resolution, indicating that crucial details may be forfeited when working at extremely low resolutions. We determine that an object inside a video frame at a specific resolution possesses sufficient information if the loss of information between that resolution and its subsequent higher resolution is minimal.

The input image is captured at N distinct resolution levels, ranging from R_0 to R_N ($R_0 < R_N$). The process begins at the lowest resolution level, R_0. The information loss, measured through the Peak Signal-to-Noise Ratio (PSNR), between the images at R_0 and R_1 is then computed. If the PSNR value is below a predefined threshold, $PSNR_{Th}$, the analysis proceeds to R_1, and compares its PSNR with that of R_2. This stepwise progression continues until the optimal resolution level, R_{opt}, is identified.

The underlying rationale for this approach is that the mutual information between consecutive resolution levels will decrease as the resolution is reduced. This is expressed as PSNR(R_0, R_1) < PSNR(R_{N-1}, R_N). However, determining the optimal resolution based on PSNR levels necessitates defining an appropriate PSNR threshold. This is done using the training dataset. Algorithm 1 outlines the procedure for establishing the PSNR threshold.

Algorithm 1 PSNR Threshold Tuning

```
 1: for I, Y in Training set do
 2:     $R \leftarrow R_0$        ▷ $R_0$ refers to the lowest resolution
 3:     $I_0$ = Downscale(I,$R_0$)
 4:     while $R < R_{org}$ and $\tilde{Y} \neq Y$ do
 5:         $R_{i+1} \leftarrow R_i + \Delta_R$
 6:         $I_{R_i}$ = Downscale(I,$R_i$)
 7:         $\tilde{Y}$ = ADARE-HD($I_{R_i}$)
 8:     end while
 9:     P ← PSNR($I_{R_{i+1}}, I_{R_i}$)
10: end for
11: return $P_{mean}$, $P_{std}$
```

For each element in the training set (line 1), we first **Downscale** the object inside the video frame I to R_0, which corresponds to the lowest resolution level (lines 2-3). Subsequently, we upsample the image I by a resolution increment of $R_i + \Delta_R$ (line 5) and predict its label using **ADARE-HD** (line 6). This terminates when the resolution reaches the original image resolution, R_{org}, or when the object label is predicted correctly (line 4). We then calculate the PSNR between the images at the last recorded resolution and the one preceding it (line 9). Finally, we compute the mean and standard deviation of the recorded PSNR values across the dataset (line 11). The PSNR threshold for resolution adaptation on the test dataset is defined as follows (Eq.1):

$$PSNR_{Th} = PSNR_{mean} + 2.PSNR_{std} \tag{1}$$

where $PSNR_{std}$ is multiplied by 2 to target the 2-sigma Gaussian right tail.

C. ADARE-HD: HDC Classification

The ADARE-HD classification encoding module consists of a CNN based feature extractor followed by an RBF kernel encoder.

1) ADARE-HD: CNN Feature Extractor: Hyperdimensional Computing is known to allow lightweight and efficient classification. However, for it fails extracting relevant features when it comes to image data. To help the HDC system retrieve those feature we desgined a compact feature extractor derived from FasterRCNN [27] pre-trained on MSCOCO [28] and composed of 64 convolutional layer followed by a frozen batch normalization layer and one maxpooling layer.

2) ADARE-HD: RBF Encoding: In this section, we present a kernel-based hyperdimensional computing encoding scheme inspired by the notion that data not separable in linear space may be separable in high-dimensional nonlinear space [29]. Let the function $K(x, y) = \Phi(x).\Phi(y)$ denote the dot product of x and y in a high-dimensional space acquired by the projection function Φ. Previous research [23] has demonstrated that the inner product can approximate the RBF Function, where $K(x, y) = \Phi(x).\Phi(y) \approx z(x).z(y)$ The Gaussian kernel can now be approximated using the dot product of two functions. We opt for Fourier-based functions $z(x) \in \{cos(\omega_0.x + \psi_0), sin(\omega_1.x + \psi_1)\}$. To encode a hypervector $\mathcal{H} = h_1, h_2, ..., h_D \in \mathcal{R}^D$ using a data point in feature space $\mathbf{F} = f_1, f_2, ..., f_n$, one can employ the following encoding system:

$$h_i = \cos(\mathbf{F} \cdot \mathcal{B}_i + b) \sin(\mathbf{F} \cdot \mathcal{B}_i) \tag{2}$$

Here, $\mathcal{B}$ denotes a random basis matrix, $\mathcal{B}_i$ a column of that matrix composed $\mathcal{B}_{ij}$ elements. $\mathcal{B}_{ij} \in \mathcal{N}(0, 1)$, $\delta(\mathcal{B}i, \mathcal{B}j) = \delta ij$, and $b \in \mathcal{U}[0, 2\pi]$ where $\delta(x, y)$ refers to the cosine similarity between x and y. However, this approach [16] is memory-intensive since it requires $\mathcal{B} \in \mathcal{M}_{\mathcal{F},D}(\mathbb{R})$, representing $\mathcal{F}.D$ real numbers. Alternatively, we can replace $F.\mathcal{B}_i \forall i \in [1, D]$ with $F*B$, where * denotes the convolutional operation, and the initial F representing the data point features are filter values and the random basis B is the signal. Consequently, $B \in \mathcal{M}_{D+\mathcal{F}-1}$ requires only $\mathcal{F}+D-1$ elements. In this manner: $h = \cos(\mathbf{F} * B + b) \sin(\mathbf{F} * B)$ Replacing the matrix dot product with a convolutional operation preserves the variance hence the information inside the hypervector. This is proven in Lemma IV.1.

Lemma IV.1. *Given a hypervector* $\mathbf{H}$ *with elements* $\{h_i\}_{i=1}^D$, *a feature space* $\mathbf{F}$ *with feature vectors* $\{f_i\}_{i=1}^n$ *and a randomly generated basis matrix* $\mathcal{B} \in \mathcal{M}_{nD}(\mathbb{R})$ *where each element* $\mathcal{B}_{ij} \sim \mathcal{N}(0, 1)$ *encoding the hypervector* $\mathbf{H}$ *as in Equation 2, and a random vector* $B \in \mathcal{M}_{n+D-1}(\mathbb{R})$ *encoding the hypervector* $\mathbf{H}$ *as in Equation IV-C2. The variance of each hypervector element* h_i *can be expressed as:* $\sigma^2_{h_i} = \frac{1}{4}(1 - e^{-2\sigma^2_{F\cdot\mathcal{B}_i}})$ *where* $\sigma_{F\cdot\mathcal{B}_i}$ *denotes the standard deviation of the inner product distribution of* $\mathbf{F}$ *and* $\mathcal{B}_i$, *where each* f_i *are deterministic. Moreover, under the assumption that all* $\mathcal{B}_{ij}$ *of* B_i *are i.i.d,* $\sigma^2_{(F*B)i} = \sigma^2_{(F.\mathcal{B})_i}$

Proof Sketch. The variance of the hypervector element is given by:

$$\sigma^2_{h_i} = \mathbb{E}(h_i^2) - \mathbb{E}(h_i)^2 \tag{3}$$

we have

$$\mathbb{E}(h_i^2) = \frac{1}{4}(1 - e^{-2\sigma^2_{F.\mathcal{B}_i}}) \tag{4}$$

Given that $F.\mathcal{B} \sim \mathcal{N}(0, \sigma_{F.\mathcal{B}})$ and $b \sim \mathcal{U}[0, 2\pi]$, we have

$$\mathbb{E}^2(h_i) = 0 \tag{5}$$

From Equations 3, 4 and 5, we have: $\sigma^2_{h_i} = \frac{1}{4}(1 - e^{-2\sigma^2_{F.\mathcal{B}_i}})$

Here, σ_{h_i} represents the standard deviation of $F.\mathcal{B}_i$ or $(F\mathcal{B})i$. Given that all $\mathcal{B}_{ij}$ are i.i.d, we can derive:

$$\begin{aligned} \sigma^2_{F.\mathcal{B}_i} &= Var(\sum_{k=0}^{N} F_k.\mathcal{B}_{ik}) = \|F\|_2 \\ \sigma^2_{(F*B)i} &= Var(\sum_{k=0}^{N} F_k.B_{i-k}) = \|F\|_2 \end{aligned} \tag{6}$$

Consequently, $\sigma^2_{(F*B)i} = \sigma^2_{F.\mathcal{B}_i}$. ■

3) Hyperdimensional computing Training: During the learning phase, the HDC system recognizes recurring patterns and avoids over-saturation of class hypervectors in single-pass training [30] by adjusting the contribution of each encoded data point to the class hypervectors based on the novelty it brings. If a data point is already present in a class hypervector, HDC adds little or no data to the model to avert hypervector saturation. If the prediction aligns with the anticipated outcome, no modifications are made to prevent overfitting. Suppose we have a new training data point, $\mathcal{H}$. HDC calculates the cosine similarity between H and all class hypervectors, C_s. The similarity of this data point with class i is computed as: $\delta_i = \delta(\mathcal{H}, C_i)$. HDC updates the model according to the δ similarity. If the input data has a label l that accurately corresponds to the class, the model updates as follows $C_l \leftarrow C_l + \eta_1(1 - \delta_l) \times \mathcal{H}$ where η_1 refers to the learning rate. If the similarity between the class hypervector and the training hypervector is large (i.e, $\delta_l \approx 1$) the algorithm should retain a small part of the training hypervector. If the input l' is misclassified or very similar to the wrong class hypervector (i.e, $\delta_{l'} \approx 1$), we add the hypervector to the correct class hypervector and subtract a portion (i.e, $\delta_{l'}$) of it from the wrong class hypervector as follows: $C_{l'} \leftarrow C_{l'} - \eta_2(\delta_{l'}) \times \mathcal{H}$

D. ADARE-HD: Tracking & Tracklet label filtering

In order to conduct multi-object tracking in our research, we have employed a two-step mechanism: detection followed by tracking. The detection phase is rooted in motion analysis, as elucidated in the preceding section, while the tracking process is executed using the SORT [31] algorithm. Object association is achieved through descriptor matching, with descriptors derived from the same network responsible for extracting features in the ADARE-HD.

Although the tracking of objects relies on existing methodologies, it serves to enhance the classification of moving entities. Consider an object, A, belonging to class C_A. We initiate the tracking of object A within the scene and concurrently conduct classification using ADARE-HD. Subsequently, a classification history also called tracklet is constructed for object A. We perform a median filter on the history to eliminate abrupt change in the object classification.

The process of tracklet label filtering may be executed in one of two modes: *online* or *offline*. In the *online* mode, the median filter is applied contemporaneously to the current tracklet, refining classifications in real-time. In the *offline* mode, all tracklet labels are adjusted post-completion of the tracking task.

V. Evaluation

A. Experimental Setup

The training of the ADARE-HD model was performed on a CPU (11th Gen Intel® Core™ i7), while the testing was carried out on both FPGA and CPU platforms.

For FPGA implementation, the ADARE-HD model was first synthesized using Xilinx Vitis High-Level Synthesis (HLS) and subsequently tested with the Xilinx Vivado Design Suite. The power consumption of the ADARE-HD model, excluding the resolution adaptation module, was determined using Xilinx Vivado XPower software and directly measured on the FPGA using *PMBus (Power Management Bus)* when the resolution adaptation module was included. The FPGA employed for testing was the Xilinx Zynq UltraScale+ MPSoC ZCU104.

For the object detection and tracking task we evaluate our method on the 1^{st} track of AI City challenge [22] using the following metrics:

- (HOTA) Higher Order Tracking Accuracy [32]: Geometric mean of detection accuracy and association accuracy. Averaged across localization thresholds.
- Association Accuracy (AssA): Association Jaccard index averaged over all matching detection and then averaged over localization thresholds.
- Detection Accuracy (DetA): Detection Jaccard index averaged over localization thresholds.
- Localization Accuracy (LocA): Localization similarity averaged over all matching detections and over localization thresholds.
- ID F1 Score (IDF1):The ratio of correctly identified detection over the average number of ground-truth and computed detection.
- Mostly tracked targets (MT). It is the ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life span
- Mostly lost targets (ML) The ratio of ground-truth trajectories that are covered by a track hypothesis for at most 20% of their respective life span

In this paper, ADARE-HD was not applied to a more common multi object tracking datasets such as MOT as *the camera capturing the scene is often moving or objects are stationary*

1) AI City Challenge Truck vs Car Dataset: A variety of fast-moving road vehicles makes classification of 'truck' and 'car' in real-time a challenging task that requires certain assumptions to be made. The AI Cities Challenge policy thus follows certain rules:

- These vehicles are categorized as ”cars”: sedan cars, SUVs, vans, buses, and smaller trucks.
- These vehicles are classified as ”trucks”: medium-sized trucks such as moving trucks and garbage trucks, as well as larger trucks like tractor-trailers.

		Accuracy (%)		
		No Filtering	Tracklet Filtering	
			Online	Offline
Resolutions	R=8x8	64.65	67.0	83.13
	R=12x12	73.1	85.14	87.15
	R=16x16	73.5	86.74	90.76
	R=Ropt	72.7	85.5	91.56

TABLE I: Tracklet label filtering accuracy results

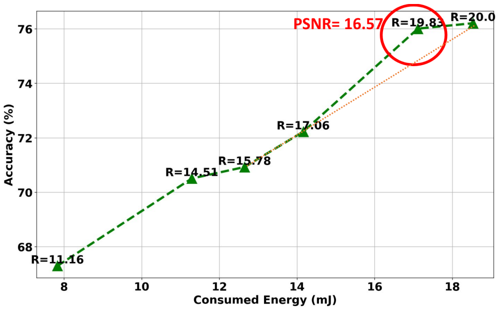

Fig. 3: Energy vs Accuracy

B. ADARE-HD Efficiency

Figure 3 presents an evaluation of the impact of selecting an optimal PSNR threshold to define the best resolution. We measure the consumed energy by the ADARE-HD and its accuracy level when applying different PSNR threshold values. A higher PSNR threshold indicates minimal information loss between two resolution levels, implying that higher resolution levels correspond to higher PSNR thresholds, and lower resolutions correspond to lower PSNR thresholds.

We incorporated the PSNR threshold selection algorithm discussed previously and determined the optimal PSNR threshold to be 16.57, corresponding to an average resolution of $R = 19.83$. The orange line in Fig. 3 represents the linear extrapolation of the accuracy curve using the adjacent resolution levels of $R = 17.06$ and $R = 20$. The accuracy at the adaptive resolution (i.e., corresponding to the PSNR threshold equal to 16.57) surpasses all values on that orange extrapolated line. This leads to the conclusion that the ADARE-HD with adaptive resolution delivers superior accuracy compared to a fixed resolution level, while maintaining the same level of energy overhead.

C. ADARE-HD & Tracklet Label Filtering

In this study, we examined the influence of tracklet-based filtering on accuracy using the AI CITY Challenge dataset [22]. We evaluated both the online and offline approaches discussed in Section IV-D, the results of which are presented in Table I.

In terms of accuracy, the offline method outperformed the online method by an average of $+8\%$. On the other hand, the online method displayed superior accuracy compared to the baseline, with an average increase of $+12.5\%$. It was seen that the tracklet label filtering introduced virtually no additional overhead. This outcome was expected, since the classification history for the offline method contains more elements, reducing the likelihood of a failure at the median filtering stage. However, it is important to recognize that choosing between the two methods is contingent upon the specific application domain — real-time or offline.

D. ADARE-HD Object Tracking Assessment

In this section we assess the detection and tracking performance of ADARE-HD on the AI CITY Challenge dataset. The main task in challenge consists of single- or multi-camera

Tracking Method	Detection network				Tracking & Detection Results						
	YOLOV3	*Mask*	*SSD*	*Motion*	*HOTA*	AssA	*MT*↑	*ML*↓	*DetA*↑	*LocA*↑	*IDF1*↑
DeepSort	✓				23.658	43.638	349	180	12.953	69.423	27.145
		✓			17.996	42.384	311	207	7.71	69.044	16.981
			✓		23.588	43.614	349	157	12.833	69.696	26.983
Tracklet	✓				23.959	44.498	388	213	13.01	69.46	27.998
		✓			19.897	42.12	468	130	9.512	69.256	20.539
			✓		25.126	**45.465**	471	142	14.006	69.841	30.057
Moana	✓				23.249	40.27	461	88	13.619	69.482	27.629
		✓			20.15	39.214	541	**41**	10.489	69.661	21.449
			✓		23.831	38.835	**571**	54	14.772	70.046	28.384
ADARE-HD (ours)				✓	**27.937**	31.52	176	279	**25.635**	**70.645**	**35.32**

TABLE II: Tracking & Detection results on AI-CITY Challenge for different tracking methods

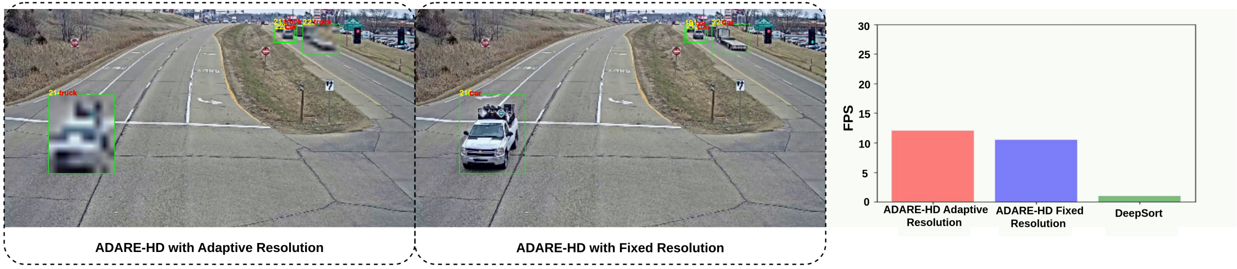

Fig. 4: Example tracking and detection results using ADARE-HD with Adaptive (left) and fixed resolutions(center). Latency overhead is shown on the right.

multi-vehicle tracking. In this paper we limited the experiments to single camera multi object tracking. Table II shows the tracking and detection results of the baselines: DeepSort [4], Tracklet [10] and Moana [19] tracking algorithms using different deep neural network (DNN) based object detection models (YOLOv3 [33], Mask-RCNN [34] and SSD-512 [35]).

ADARE-HD shows better tracking and detection performance reflected by HOTA, DetA and LocA, than the state-of-the-art. One reason is that DNN based detectors capture not only the moving object but also all static objects in the frame, while the main purpose of this challenge is tracking moving vehicles in a real-world traffic environment.

The ADARE-HD model achieves a superior ID F1 score due to its incorporation of a feature matching module that optimizes video frame objects at their optimal resolution. This finding corroborates the experimental outcomes illustrated in Figure 3, which demonstrate that optimal accuracy can be attained at lower resolutions. Furthermore, by utilizing adaptive resolution features, the distinctiveness of the video frame objects is enhanced, leading to more accurate object identification across consecutive frames (i.e, higher ID F1 score).

		ADARE-HD		DeepSort
		Adaptive Resolution	*Fixed Resolution*	
Latency (s)	Detection	0.14	**0.093**	1.02
	Tracking	**0.0014**	**0.0014**	0.0036
Energy Efficiency		**1.6**	0.75	1

TABLE III: ADARE-HD Tracking, Detection Latency Overhead and Energy Efficiency compared to state-of-the-art

The aim of the experiment shown in Table III is to illustrate the overhead, specifically latency and energy efficiency, associated with the ADARE-HD Tracking and detection algorithm in comparison to the state-of-the-art algorithm, DeepSort [4]. The evaluation is conducted on the AI-CITY Challenge test set and executed on a CPU.

As clearly depicted in Table III, the adaptive resolution ADARE-HD demonstrates significant speed, operating approximately *6.23* times faster than DeepSort [4]. This difference is largely attributed to the heavy reliance of DeepSort [4] on a DNN object detector. Moreover, ADARE-HD with adaptive resolution outperforms ADARE-HD with a fixed resolution, being roughly 33% faster compared to the case when the resolution is set at the highest value.

Utilizing the AI CITY Challenge dataset [22], we benchmarked ADARE-HD's energy efficiency against a deep learning-based method under fixed and adaptive resolutions (Table III). The adaptive variant outperformed DeepSort [4] by 1.6 times, while the fixed resolution model showed reduced efficiency since it is fixed to the highest possible resolution. The adaptive resolution module appears crucial for ADARE-HD's low-power object tracking efficacy.

Figure 4 presents an example of tracking and detection in a scene from the AI-CITY Challenge Dataset. The left figure displays the results of ADARE-HD with adaptive resolution. Here, we see that the boxes around vehicles appear blurry due to the adaptive resolution at which ADARE-HD operates. The center figure illustrates the results of ADARE-HD with fixed, high resolution. Notably, the fixed-resolution version of ADARE-HD incorrectly classifies trucks, an error that is not present in the adaptive resolution version. Finally, the right figure delineates the speed of each algorithm on a CPU, expressed in FPS.

VI. Conclusion

The proliferation of edge computing devices necessitates low-power, accurate embedded algorithms for machine learning applications. This paper presented ADARE-HD, a low-power hyperdimensional framework for multi-object detection, classification and tracking. ADARE-HD adjusts its operating resolution to balance overhead with performance, providing low-overhead and accurate tracking capabilities.

Acknowledgment

This research was supported by NSF Grant No. 2414361 and by the School of ECE at Georgia Institute of Technology.

References

[1] Yongxi Lu and Tara Javidi. Efficient object detection for high resolution images. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1091–1098, 2015.

[2] Salma Abdel Magid, Francesco Petrini, and Behnam Dezfouli. Image classification on iot edge devices: profiling and modeling. *Cluster Computing*, 23:1025–1043, 2020.

[3] Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1:139–159, 2009.

[4] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric, 2017.

[5] Joshua W. Wells and Abhijit Chatterjee. Content-aware low-complexity object detection for tracking using adaptive compressed sensing. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 8(3):578–590, 2018.

[6] Wei Zhang, Gregory Zelinsky, and Dimitris Samaras. Real-time accurate object detection using multiple resolutions. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[7] Dennis Park, Deva Ramanan, and Charless Fowlkes. Multiresolution models for object detection. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 241–254.

[8] Zhenhua Huang, Shunzhi Yang, MengChu Zhou, Zheng Gong, Abdullah Abusorrah, Chen Lin, and Zheng Huang. Making accurate object detection at the edge: review and new approach. *Artificial Intelligence Review*, 55(3):2245–2274, 2022.

[9] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2016.

[10] Jinlong Peng, Tao Wang, Weiyao Lin, Jian Wang, John See, Shilei Wen, and Erui Ding. Tpm: Multiple object tracking with tracklet-plane matching. *Pattern Recognition*, 107:107480, 2020.

[11] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. A multi-task deep network for person re-identification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017.

[12] Mingjian Zhu, Kai Han, Enhua Wu, Qiulin Zhang, Ying Nie, Zhenzhong Lan, and Yunhe Wang. Dynamic resolution network, 2021.

[13] Neil C Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F Manso. The computational limits of deep learning. *arXiv preprint arXiv:2007.05558*, 2020.

[14] Cheng-Yang Chang, Yu-Chuan Chuang, Chi-Tse Huang, and An-Yeu Wu. Recent progress and development of hyperdimensional computing (hdc) for edge intelligence. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 13(1):119–136, 2023.

[15] Arpan Dutta, Saransh Gupta, Behnam Khaleghi, Rishikanth Chandrasekaran, Weihong Xu, and Tajana Rosing. Hdnn-pim: Efficient in memory design of hyperdimensional computing with feature extraction. In *Proceedings of the Great Lakes Symposium on VLSI 2022*, pages 281–286, 2022.

[16] Yang Ni, Nicholas Lesica, Fan-Gang Zeng, and Mohsen Imani. Neurally-inspired hyperdimensional classification for efficient and robust biosignal processing. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, New York, NY, USA, 2022. Association for Computing Machinery.

[17] Mohsen Imani, Chenyu Huang, Deqian Kong, and Tajana Rosing. Hierarchical hyperdimensional computing for energy efficient classification. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pages 1–6, 2018.

[18] Hai Wu, Qing Li, Chenglu Wen, Xin Li, Xiaoliang Fan, and Cheng Wang. Tracklet proposal network for multi-object tracking on point clouds. In *IJCAI*, pages 1165–1171, 2021.

[19] Zheng Tang and Jenq-Neng Hwang. Moana: An online learned adaptive appearance model for robust multiple object tracking in 3d. *IEEE Access*, 7:31934–31945, 2019.

[20] Ashwani Aggarwal, Susmit Biswas, Sandeep Singh, Shamik Sural, and Arun K Majumdar. Object tracking using background subtraction and motion estimation in mpeg videos. In *Computer Vision–ACCV 2006: 7th Asian Conference on Computer Vision, Hyderabad, India, January 13-16, 2006. Proceedings, Part II 7*, pages 121–130. Springer, 2006.

[21] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking, 2016.

[22] Milind Naphade, Shuo Wang, David C. Anastasiu, Zheng Tang, Ming-Ching Chang, Yue Yao, Liang Zheng, Mohammed Shaiqur Rahman, Archana Venkatachalapathy, Anuj Sharma, Qi Feng, Vitaly Ablavsky, Stan Sclaroff, Pranamesh Chakraborty, Alice Li, Shangru Li, and Rama Chellappa. The 6th ai city challenge, 2022.

[23] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007.

[24] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. Ieee, 2001.

[25] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7464–7475, 2023.

[26] Chaohui Zhan, Xiaohui Duan, Shuoyu Xu, Zheng Song, and Min Luo. An improved moving object detection algorithm based on frame difference and edge detection. In *Fourth international conference on image and graphics (ICIG 2007)*, pages 519–523. IEEE, 2007.

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks.

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context.

[29] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc.

[30] Alejandro Hernández-Cano, Namiko Matsumoto, Eric Ping, and Mohsen Imani. Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 56–61, 2021.

[31] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2016.

[32] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129:548–578, 2021.

[33] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[34] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2018.

[35] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *Computer Vision ECCV 2016*, pages 21–37. Springer International Publishing, 2016.