HOGEye: Neural Approximation of HOG Feature Extraction in RRAM-Based 3D-Stacked Image Sensors

Tianrui Ma[1], Weidong Cao[1], Fei Qiao[3], Ayan Chakrabarti[2], Xuan Zhang[1]
[1]Department of ESE and [2]Department of CSE, Washington University in St. Louis, St. Louis, Missouri, USA;
[3]Department of Electronics Engineering, Tsinghua University, Beijing, China;
{tianrui.ma,weidong.cao,ayan,xuan.zhang}@wustl.edu;qiaofei@tsinghua.edu.cn.

ABSTRACT

Many computer vision tasks, ranging from recognition to multi-view registration, operate on feature representation of images rather than raw pixel intensities. However, conventional pipelines for obtaining these representations incur significant energy consumption due to pixel-wise analog-to-digital (A/D) conversions and costly storage and computations. In this paper, we propose HOGEye, an efficient near-pixel implementation for a widely-used feature extraction algorithm—Histograms of Oriented Gradients (HOG). HOGEye moves the key but computation-intensive derivative extraction (DE) and histogram generation (HG) steps into the analog domain by applying a novel neural approximation method in a resistive random-access memory (RRAM)-based 3D-stacked image sensor. The co-location of perception (sensor) and computation (DE and HG) and the alleviation of A/D conversions allow HOGEye design to achieve significant energy saving. With negligible detection rate degradation, the entire HOGEye sensor system consumes less than $48\mu W$@30fps for an image resolution of 256×256 (equivalent to $24.3pJ/pixel$) while the processing part only consumes $14.1pJ/pixel$, achieving more than 2.5× energy efficiency improvement than the state-of-the-art designs.

CCS CONCEPTS

• **Hardware → On-chip sensors**.

KEYWORDS

HOG, RRAM, Neural Approximation, Near Pixel Processing

ACM Reference Format:
Tianrui Ma[1], Weidong Cao[1], Fei Qiao[3], Ayan Chakrabarti[2], Xuan Zhang[1]. 2022. HOGEye: Neural Approximation of HOG Feature Extraction in RRAM-Based 3D-Stacked Image Sensors. In *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED '22), August 1–3, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3531437.3539706

1 INTRODUCTION

Extracting high-dimensional features of an object is a primary step in object detection and it has been extensively studied in the field of computer vision [15]. Efficient extraction of features is critical when deploying real-time perception onto resource-constrained platforms. Despite recent advancements of learning features from deep neural network (DNN) models, hand-crafted features exhibit several orders of magnitude higher energy efficiency in actual hardware implementation and remain a compelling design choice for energy-critical systems [25]. We focus on Histogram of Oriented Gradients (HOG) algorithm, which is widely used to extract hand-crafted spatial features of an image. HOG algorithm can achieve high detection rate and effective geometric extraction with relatively simple computations, making it attractive for applications ranging from automated surveillance for environmental monitoring to obstacle avoidance and simultaneous-localization-and-mapping (SLAM) for autonomous micro-robotics [14].

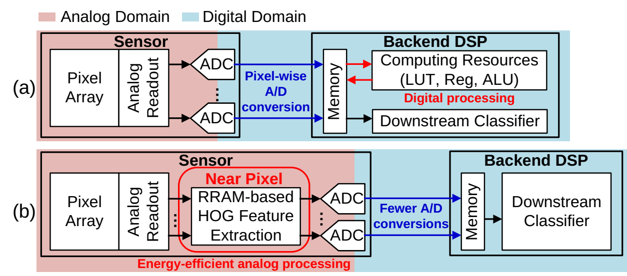


Figure 1: (a) Conventional digital HOG implementation. (b) Proposed HOGEye sensor system.

Conventional HOG implementation suffers from high energy consumption due to pixel-wise analog-to-digital (A/D) conversion and the resulting costly resource usage in digital processing. As Fig. 1(a) illustrates, the image sensor quantizes all pixels and the digital backend consumes large number of registers, look-up tables (LUTs) and arithmetic-logic units (ALUs) to extract HOG features, causing significant energy overheads [11].

To address these issues, we propose HOGEye – an energy-efficient near-pixel processing architecture to extract HOG features in the analog domain before A/D converter (ADC). As shown in Fig. 1(b), the analog readouts from the pixel array are directly used for HOG feature extraction. The generated analog HOG features are then quantized by ADC and processed by the digital backend where downstream classifier resides. At circuit level, HOGEye adopts resistive-random-access-memory (RRAM)-based neural approximation to efficiently realize the function of spatial derivative calculation and facilitate histogram generation. At architecture level, HOGEye takes the advantage of three-dimensional (3D) heterogeneous integration technology and is modeled after a two-layer stacked sensor system where pixel substrate (pixel array) stacks on top of processor substrate (processing circuitry) in the sensor.

ISLPED '22, August 1–3, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9354-6/22/08.
https://doi.org/10.1145/3531437.3539706

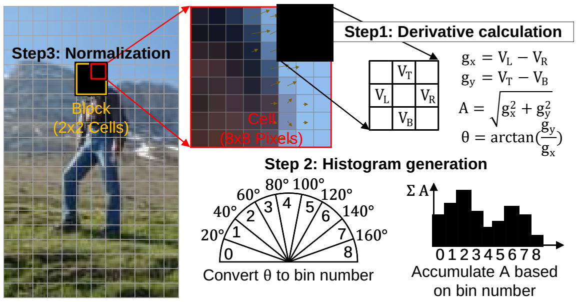


Figure 2: Illustration of HOG algorithm.

Compared to the conventional HOG implementation, on the processor substrate, HOGEye reduces the number of A/D conversions by 7.1×, and consumes only 14.1$pJ/pixel$ without extra area overhead, outperforming state-of-the-art mixed-signal [6] and digital [26] HOG implementation by more than 2.5× and 11×, respectively.

2 BACKGROUNDS

2.1 Histogram of Oriented Gradients (HOG)

HOG algorithm is based on evaluating normalized histograms in local regions of an image, and the essential steps are shown in Fig. 2. First, each pixel's horizontal gradient g_x and vertical gradient g_y are obtained by linear difference of neighbor pixels. With g_x and g_y, the pixel's full spatial derivative is obtained in the form of magnitude A and orientation θ (Step 1). Second, for all the spatial derivatives within the non-overlapping 8×8 pixels (called a *cell*), their orientations are discretized to one of the nine bin numbers, and their magnitudes are distributed into the corresponding bin according to the orientation bin number. Each cell thus generates a 1×9 vector, known as a *histogram* (Step 2). Third, for all the histograms within the overlapping 2×2 cells (called a *block*), normalization is performed to eliminate shadow effect. Each block thus generates a $(1 \times 9) \times 4 = 1 \times 36$ HOG vector (Step 3). HOG vectors from different blocks are concatenated sequentially to construct the final HOG features of the entire image.

HOG feature is proven to be an effective local descriptor with high accuracy in human detection, outperforming Harr wavelets, scale-invariant feature transform, Gabor filters, and shape contexts [7]. However, HOG feature extraction is computationally intensive and time-consuming, especially the spatial derivative calculation (Step 1) and histogram generation (Step 2). In this paper, HOGEye aims to implement these two parts while leaving the block normalization (Step 3) to the digital backend.

2.2 Existing HOG Implementations

HOG algorithms are conventionally implemented in FPGA or digital ASIC. In both cases, pixel-wise A/D conversions at the frontend and massive digital data movements at the backend are unavoidable. As for HOG feature extraction, the FPGA design [11] takes tens of thousands of LUTs and registers, and the digital ASIC design [25] takes 893k gates, causing significant power and area overheads. Recent works also explore analog HOG implementations. For example, digital horizontal/vertical gradient can be converted to the analog domain for orientation binning, saving 93% of area as compared to the digital counterpart [6]. However, this method suffers from energy-intensive pixel-wise A/D and D/A conversions. Besides, logarithm horizontal/vertical gradient is proposed to replace linear gradient to realize data compression in both sensor frontend and digital backend [28]. However, only one-dimensional gradients are generated from sensor rather than the full histograms, causing the overhead of histogram generation at the backend.

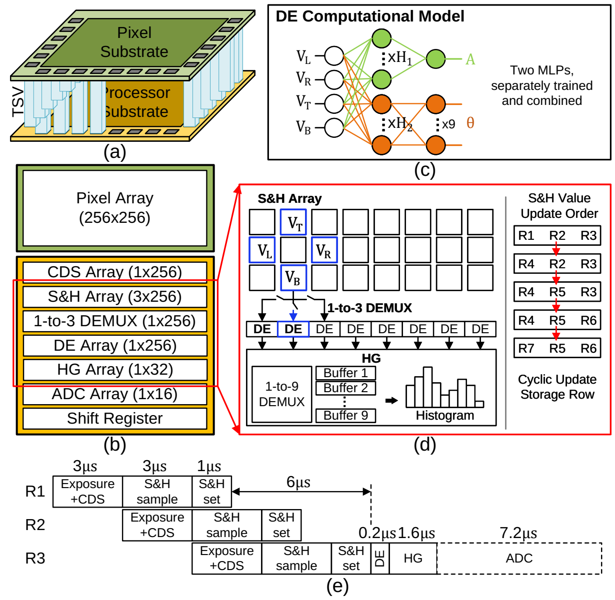


Figure 3: Overview of HOGEye. (a) 3D structure, (b) substrate floorplan, (c) DE computational model, (d) computation signal flow, and (e) system processing sequence.

2.3 Sensing-Processing With 3D-Stacked Sensor

Modern three-dimensional large-scale integration (3D-LSI) technology allows different substrates of heterogeneous process to be stacked and connected by through-silicon-vias (TSVs). As predicted by 3D-LSI roadmap, the pitch of TSV can keep shrinking to as narrow as $1\mu m$, paving the way for high performance 3D-stacked sensor design [21]. Sony proposes the industry's first three-layer stacked sensor with "pixel-DRAM-logic" structure [10]. Pixel values from pixel substrate are transferred to logic substrate for quantization, and to DRAM substrate for digital storage, so as to improve frame rate. The connections between substrates are realized by 35k TSVs whose size and pitch are $2.5\mu m$ and $6.3\mu m$, respectively. Recently Sony implements a digital CNN processor below the pixel substrate [12], making sensing-processing system a lightweight frontend node. Many other works also explore integrating complicated machine learning engine with the advantage of 3D-stacked sensor architecture (e.g. CAMEL [8]).

3 PROPOSED HOGEYE SENSOR SYSTEM

3.1 System Overview

The proposed HOGEye sensor performs both sensing and HOG histogram computing. It adopts the structure of 3D stacked sensor by implementing the two functions with two TSV-connected substrates: a pixel substrate and a processor substrate (Fig. 3(a)).

As shown in Fig. 3(b), the pixel substrate includes a 256×256 pixel array. The processor substrate includes column-parallel correlated-double-sampling (CDS) array, 3×256 sample-and-hold (S&H) array, column-parallel derivative extractor (DE) array, 8-column-shared histogram generator (HG) array, 16-column-shared ADC array as well as peripheral switches and readout registers.

Analog pixel values are obtained by the pixel substrate, and sent to the processor substrate through TSVs for analog buffering and computing. 256 TSVs are needed to transfer a row of pixel values in parallel. As shown in Fig. 3(d), S&H array holds three rows of pixel values, and each DE takes four pixel values around a central pixel and calculates the central pixel's derivative. The calculated derivatives from 8 DEs are moved to the same HG to generate a partial histogram with 9 bins. After processing 8 rows of pixels, each HG contains the complete histogram from a cell (referring to Sec. 2.1), then ADC quantizes the histogram bin by bin. Since each cell only needs 9 quantizations, the number of A/D conversions is reduced by 8×8/9≈7.1 times as compared to pixel-wise quantization in conventional methods.

Fig. 3(e) shows that HOGEye sensor works in sequential manner as "sensing → computation → readout" while in the sensing stage it performs "exposure → analog buffering" in pipeline manner. The computation will not start until the sensing of the third pixel row is finished. To accommodate with the latency of pixel exposure and CDS ($3\mu s$), S&H sampling time is set to $3\mu s$. The pipeline mainly helps reduce S&H holding time, holding energy, and analog signal degradation.

3.2 Analog Buffering

The right part of Fig. 3(d) shows the operation of analog buffering. As the pixel substrate performs exposure in rolling shutter way, 3 rows of pixels (R1-R2-R3) are read out row by row and stored at the S&H array temporarily for later processing. After computing the partial histogram from one row of derivative, a new pixel row (R4) is stored to the S&H array, taking up the place of pixel row R1; then after another processing cycle, pixel row R5 is stored to replace pixel row R2. In this cyclic way, only one row is updated during each S&H update cycle so as to maximize circuit reuse. Since derivative calculation requires overlapped input pixels, a 1-to-3 analog DEMUX is placed at the end of every S&H column such that every stored value has access to three adjacent DEs.

3.3 Neural-Approximated Derivative Extraction

Universal approximation theorem proves that a 3-layer multi-layer-perceptron (MLP) with continuous nonlinear activation function can be trained to accurately approximate arbitrary input-output relationship [13]. Prior works show the potential of this theorem on approximating analog computing [5, 17] (analog input to analog output) and A/D conversion [2, 3] (analog input to digital output). Therefore, the similar strategy can be promisingly applied for HOG feature extraction, which has analog input (analog pixel values) and mixed-signal output (spatial derivative's analog magnitude A and digital orientation θ). Compared to explicit non-linear derivative function, the strategy enables the proposed DE to perform only linear vector-matrix-multiplication (VMM) and non-linear activation operations, thereby making the DE suitable to be implemented on RRAM crossbar array.

The MLP model has 4 inputs (V_L, V_R, V_T, V_B), 10 outputs, and H_1+H_2 hidden neurons, as shown in Fig. 3(c). The subscript in the inputs represents left, right, top and bottom to the central pixel. The outputs include **one** normalized analog value representing A ($A \in [0,1]$), and **nine** digits that encodes θ to 9-bit one-hot code, representing 9 bins. To classify the continuous θ ($\in [0,\pi]$) to discrete bins, we compare g_y with $g_x \cdot \tan\frac{\pi}{9}i$ to avoid explicit division and inverse trigonometric function. The objective function is then defined as:

$$A = \sqrt{\frac{g_x^2 + g_y^2}{2}}, \quad \text{index}(\theta) = \text{Compare}(g_y, g_x \tan\frac{\pi}{9}i) \tag{1}$$

where $i \in [1,9]$. For example, θ belongs to 2^{nd} bin if: $g_x \tan(\frac{\pi}{9}) < g_y < g_x \tan(\frac{2\pi}{9})$. A and θ are trained separately by two MLPs, which share the same inputs but have exclusive hidden neurons and outputs. Mean-squared-error and cross-entropy are used as loss function to train A and θ, respectively.

4 IMPLEMENTATION OF SYSTEM BUILDING BLOCKS

4.1 Pixel, TSV and S&H Circuit

Fig. 4 shows the circuitry of pixel and interface between pixel and DE. The sensor uses typical 4-transistor active pixel. The pixel output goes to bond pad on the pixel substrate, flows through TSV, and reaches to bond pad on the processor substrate. Compared to direct connection between circuit blocks, hiding TSV under bond pad allows larger TSV pitch and lower fabrication cost [21]. Here TSV is chosen to have $2\mu m$ size and $20\mu m$ length with 0.1Ω resistance and 52fF capacitance [16]. CDS samples pixel's readout before and after exposure sequentially, and conducts subtraction to reduce fixed pattern noise. Referring to system processing sequence in Fig. 3(e), each S&H unit needs to hold value for at least $6\mu s$ so a low leakage S&H design with negative feedback is chosen [23]. CDS and S&H construct the analog interface between pixel and the following processing unit, avoiding digital interface in conventional sensors [19].

4.2 Derivative Extractor (DE)

Fig. 5(a) shows the DE circuitry that performs VMM, analog shift-and-add (S+A), and non-linear activation (NAF). VMM is realized by RRAM crossbar array, where each RRAM cell is a 1T1R structure consisting of one switch transistor and one RRAM device (Fig. 5(b)). For simplicity, the transistor in the RRAM cell is omitted in Fig. 5(a).

The trained weights are mapped to the conductance of RRAM devices. To deal with weight polarity, we adopt differential RRAM array structure with complementary input proposed in the previous work [3]. In this strategy, positive and negative weight matrices are mapped to positive and negative path, respectively. Inside each path both upper and lower RRAM sub-arrays are used to receive complementary inputs ($\vec{x}^p$, $\vec{x}^n$) interchangeably. In Fig. 5, the positive path of Layer 1 contains upper RRAM sub-array (blue-shadowed) and lower RRAM sub-array (yellow-shadowed). The upper sub-array receives $\vec{x}^p$=[V_{DD}, V_T, V_B, V_L, V_R] as input while the lower one receives $\vec{x}^n$=V_{DD} - $\vec{x}^p$=[0, V_T', V_B', V_L', V_R'] where V_{DD} and 0 are used to map bias. Output current of each RRAM column is the weighted sum of input with corresponding RRAM conductances:

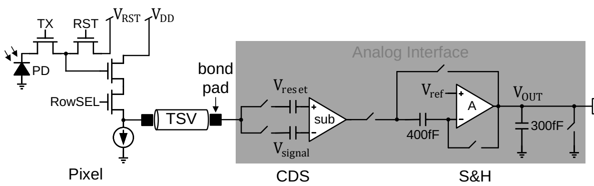


Figure 4: Sensing-processing interface circuit between the two substrates.

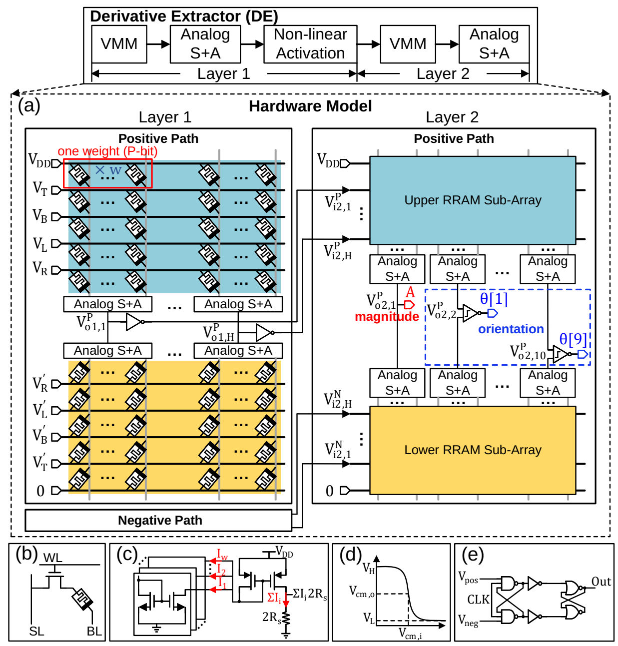


Figure 5: Circuit of (a) DE, (b) RRAM cell, (c) analog S+A, and (e) comparator[27]. (d) illustrates CMOS inverter voltage transfer curve.

$I_j = \sum_{i=1}^{5} x_i g_{i,j}$. The negative path has the same structure except that the positions of complementary input vectors are exchanged.

One challenge here is the limited precision of RRAM device (2-4 bit) [4], so we map a high precision weight (P-bit) into multiple low-precision RRAM devices. We use w RRAM devices to represent a single weight, with each device bearing the precision of $\frac{P}{w}$-bit. To accumulate currents from w RRAM columns, analog S+A implemented by ratioed current mirrors (Fig. 5(c)) adds the currents up with binary-weighting as $I^{sum} = \sum_{j=1}^{w} 2^{\frac{P}{w}(j-1)} I_j$. For example, if P=6-bit and w=3, then one 6-bit weight is instantiated onto three 2-bit RRAM devices, whose conductance are w_2, w_1 and w_0. The final sum is: $2^4 w_2 \cdot x + 2^2 w_1 \cdot x + 2^0 w_0 \cdot x$, with the current mirrors' ratios are 16, 4, 1, respectively. The accumulated current is converted to voltage via a sampling resistor $2R_s$ and received by a CMOS inverter emulating sigmoid function [3] (Fig. 5(d)). Combining the two sub-arrays, input voltage to the h^{th} inverter is derived as:

$$\begin{aligned} V^{p}_{o1,h} &= (I^{sum,u}_{h} + I^{sum,l}_{h}) \times (2R_s || 2R_s) \\ &= R_s \Big(\sum_{j=1}^{w} 2^{r(j-1)} \sum_{i=1}^{5} x^{p}_{i} g^{u}_{h,i,j} + \sum_{j=1}^{w} 2^{r(j-1)} \sum_{i=1}^{5} x^{n}_{i} g^{l}_{h,i,j}\Big) \end{aligned} \tag{2}$$

where $r = \frac{P}{w}$ and superscript u and l represent upper and lower sub-array. Same derivation can be applied to Layer 1's negative path to obtain $V^{n}_{o1,h}$ and Layer 2 to obtain $V^{p}_{o2,h}$. Note that Layer 2's negative path is not necessary because the positive path is already the final output. Before each θ output port there is a three-input NAND gate comparator [27] (Fig. 5(e)) for binarization.

Note that the RRAM conductance $g_{h,i,j}$ in the above derivation has no linear projection to the trained weight, because during mapping the trained weights are normalized to guarantee that the conductance is positive and fall into a reasonable range [3]. Therefore, compared with the absolute output value, the output voltage represented by the mapped conductance (Eq. (2)) has a scaling factor F and a constant offset $\frac{V_{DD}}{2}$. F is related to the sum of conductance placed in the same column, thus it is unique under different mappings. We will explain how to deal with F in Sec. 5.2.

4.3 Histogram Generator (HG) and ADC

HG contains an 1-to-9 analog DEMUX and 9 analog buffers. The calculated derivative magnitude (in current domain) is charged to one of the 9 analog buffers according to its orientation bin number. Each analog buffer is a switch-controlled capacitor, including one metal-insulator-metal capacitor (500fF) and two low leakage switches used in [23] for read/write and reset. One HG receives the magnitudes from 8 DEs sequentially. After processing 8 rows of pixels, 64 (8×8) magnitudes are accumulated at each HG, then the voltage at each bin is quantized by ADC sequentially. We use the SAR ADC design in [9] with sampling rate of 2.5MS/s. To reduce ADC's area overhead and utilize its high sampling rate, one ADC is shared by 2 HGs, meaning that one ADC needs to convert 18 (2×9) values sequentially.

5 EVALUATION

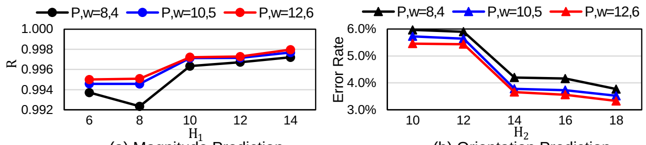
5.1 Evaluation Methodology

Simulation. First, we train the neural-approximated DE with TensorFlow, and test inference accuracy of spatial derivative extraction with hardware noise injection. Second, we generate SPICE netlist by instantiating the trained weights with HfO_x-based RRAM model, and conduct circuit simulation with CMOS building blocks (S&H, analog S+A, NAF and HG, in 130nm) in Cadence Virtuoso to evaluate energy consumption. Third, we validate the learned HOG features on DaimlerChrysler [22] dataset for task-level detection rate with hardware noise injection. We take 1,000 positive/negative samples from the dataset, with 800 for training and 200 for testing, and use linear support-vector-machine (SVM) as classifier. During system design space exploration, design parameters include the number of hidden neurons (H), weight precision (P) and weight split-level (w). These parameters determine the DE's inference accuracy and energy consumption, thereby affecting the entire HOGEye sensor's task-level accuracy and energy efficiency.

Noise model. Main hardware noise includes sensing noise (shot noise, read noise) and processing noise (RRAM resistance variation), and we model them as Gaussian disturbance (σ_1, σ_2) in input voltages of RRAM array x_i and RRAM device conductance g_i, respectively:

$$\tilde{x}_i = N(x_i, \sigma_1^2 = f_1 f_2 x_i + f_2^2 \sigma_r^2), \quad \tilde{g}_i = N(g_i, \sigma_2^2) \tag{3}$$

where f_1, f_2, σ_r are sensor's conversion gain, column gain and read noise [1] and we conservatively set them to $100\mu V/e^-$, 1 and $10mV$;



(a) Magnitude Prediction (b) Orientation Prediction

Figure 6: Pearson correlation R and error rate for magnitude and orientation prediction under different (H, P, w).

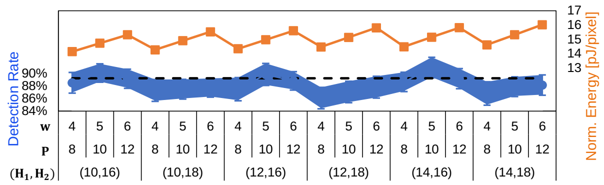


Figure 7: Under different (H, P, w): averaged detection rate (blue), and normalized energy consumption of HOGEye's processing part (orange).

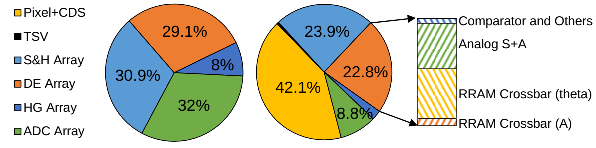
and $\sigma_2 = 0.015 g_i$ as reported in [20]. Substituting Eq. (3) to Eq. (2) gives the noise-injected output voltages.

Specifically, to inject RRAM variation into trained weights, we conduct following steps: 1. Normalize and round the weights, then convert them to P-bit binary-weighted representation; 2. Split the P-bit representation to w weight levels. For each weight level, convert it to its integer representation and add Gaussian disturbance as Eq. (3); 3. Multiply each noise-injected weight level with corresponding binary-weighting factor and add all weight levels together, which gives weights injected with RRAM variation.

5.2 Accuracy Validation

DE accuracy. To explore how the design parameters affect DE's inference, we evaluate the prediction accuracy for A and θ against different (H, P, w). For conservative estimation, we map trained weights onto 2-bit RRAM devices by limiting P/w to 2. We use two metrics to measure prediction accuracy for A and θ, due to their difference in datatype (analog vs. digital) and the property of our mapping method. For A prediction, as stated in Sec 4.2 the circuit output has a scaling factor F that changes along with mapped conductance, thus using circuit output $V^{p}_{o2,1}$ to recover real magnitude is difficult. However, under the same mapping, F keeps uniform so that the specific value of F does not affect the generated histogram's shape. And for HOG algorithm, it is the histogram's shape rather than absolute value that matters, so we use Pearson correlation coefficient R between circuit output and magnitude groundtruth as the metric (Fig. 6(a)) . Higher R means higher linearity between circuit output and groundtruth, as well as more accurate circuit output. Increasing H_1 from 6 to 14, the coefficient R increases towards 1, with the trend being stable around $H_1 = 10$. For θ prediction, we use binning error rate as the metric (Fig. 6(b)). One-hot θ is obtained by comparing circuit outputs with the offset $\frac{V_{DD}}{2}$, and it is not skewed by the scaling factor F. Increasing H_2 from 10 to 18, the error rate decreases from 6% to nearly 3%, with the trend being stable around $H_2 = 14$.

System detection rate. Based on Fig. 6, we choose H_1 from {10, 12, 14} and H_2 from {16, 18} as these sets show the highest prediction accuracy. Together with different (P, w), we evaluate task-level



(a) Area (processor only) (b) Energy (pixel+processor)

Figure 8: (a) Area and (b) energy consumption breakdown of HOGEye sensor under (H, P, w)=(10+16, 8, 4). "Others" includes energy of RRAM read, write, leakage, reset and set.

detection rate with the trained DE array on DaimlerChrysler dataset. First, we inject sensing noise to training set and train a linear SVM. Second, we convert images in test set to voltages (with sensing noise) as DE array's input, add RRAM variation, and send the output histograms to the pre-trained SVM for detection rate. Third, to overcome stochastic property in the output, we perform 50 tests at each (H, P, w), and plot averaged detection rate with standard deviation as error band in Fig. 7 (blue). Compared with baseline (black dotted line), HOGEye achieves less than 1% detection rate degradation in most cases.

5.3 System Hardware Performance

Area overhead. For the pixel substrate, we set pixel resolution to be 256×256 and pixel pitch to be $5\mu m \times 5\mu m$, so pixel array area is $1.64 mm^2$. For the processor substrate, in the worst case, we set $(H_1 + H_2) = (14 + 18)$, then less than 256KB RRAM cells are used, which have the area of $0.51 mm^2$ according to DESTINY[24] simulator. Besides, S&H array, CMOS inverters in DE array, HG array and ADC array consume about $0.54 mm^2$, $200 \mu m^2$, $0.14 mm^2$, and $0.56 mm^2$, respectively. Therefore, the two substrates have comparable area. The estimated area breakdown for the processor substrate is shown in Fig. 8(a).

Energy consumption. The energy breakdown for the whole sensor under parameter $(H_1 + H_2, P, w) = (10 + 16, 8, 4)$ is shown in Fig. 8(b). ADC energy is significantly reduced by analog computing. Zooming into DE array's energy, the neural approximator for A costs less than the neural approximator for θ due to fewer hidden/output neurons. Since pixel array does not participate any HOG computation, it is excluded from sensor's energy estimation in the following discussion. To explore the effect of different parameters on HOGEye sensor's energy, the trend of normalized energy is plotted in Fig. 7 (orange). With (H_1, H_2) increasing, the energy increases due to larger number of inverters in RRAM crossbar array. And under each (H_1, H_2), the energy increases along with (P, w) due to larger number of analog S+A.

5.4 Performance Comparison and Discussion

We compare our work with one mixed-signal design [6], one digital design [26] and one analog design [28] in Table 1. We conservatively set HOGEye's frame rate to 30fps to be comparable with other works, though higher frame rate can be achieved. For FoM_1 which normalizes energy consumption by pixel array resolution, HOGEye achieves 2.7×, 33.4× and 5.9× higher efficiency than [6], [26] and [28]. However, FoM_1 does not consider the extracted feature's complexity. For example, in the digital design [26], they extract 12-level HOG features from a single frame. Thus we

Table 1: HOGEye Specifications and Comparison with State-of-the-art

	Our Work	JSSC [6]	JSSC [26]	JSSC [28]
Process	130nm CMOS + RRAM	180nm CMOS	65nm CMOS	130nm CMOS
Pixel Array	256×256	256×256	1920×1080	320×240
Supply Volt.	1.5V	0.8V	0.77V	1.5V/0.9V
Frame Rate	30fps	15fps	30fps	30fps
Feature Type	single-scale histograms	single-scale histograms	multi-scale HOG	multi-scale log-gradient
Imple-mentation	analog	mixed-signal	digital	analog
CR@A/D Interface*	$\frac{1}{7.1}$	1	-	$\frac{1}{5.3} \sim \frac{1}{2.9}$
Number of Histograms	1024	1024	87188	-
FoM_1^{**} [pJ/pixel]	**14.1~16***	38.5	471	83.7~97.9
FoM_2^{**} [nJ/hist.]	**0.9~1***	2.5	11.2	-

* compression ratio (CR) = $\frac{\text{sensor generated digits}}{\text{pixel resolution}\times\text{8bit}}$.

** $\text{FoM}_1 = \frac{\text{sensor energy (excluding pixel array)}}{\text{sensor resolution}}$, $\text{FoM}_2 = \frac{\text{sensor energy (excluding pixel array)}}{\text{number of histograms generated}}$.

*** Our results come from simulation, while the results of other works come from real chip measurement.

additionally compare FoM_2, which normalizes the energy consumption by the number of HOG histograms generated from single frame. For FoM_2, HOGEye achieves 2.5× and 11× higher energy efficiency than [6] and [26]. Besides, compared with [28] where compressive log-gradient is generated, HOGEye achieves higher compression ratio because denser histogram is generated. Finally, as a quantitative comparison of the energy overhead between DNN learned features and hand-crafted features, RedEye [18] consumes $3.3nJ/pixel$ when implementing only one convolutional layer, which is 200× higher than the proposed HOGEye sensor. It can be estimated that learning features using DNN with more layers must consume more energy.

In this paper we only discuss HOG algorithm implementation. However, HOGEye sensor architecture can be applicable to various block-based image processing algorithms, such as block-based learning/compression and compressive sensing, depending on different MLP (or more advanced neural network) structure and RRAM crossbar array configuration. We leave generalizing HOGEye sensor architecture as future work.

6 CONCLUSION

We propose HOGEye, an efficient near-pixel processing architecture for HOG feature extraction in 3D-stacked image sensor. Spatial derivatives and histograms are generated in the analog domain with raw pixel values via column-parallel RRAM-based neural-approximated derivative extractors and column-shared histogram generators, reducing required A/D conversions by 7.1×. HOGEye's processing part only consumes $14.1pJ/pixel$ with less than 1% detection rate degradation, which is more than 2.5× and 11× better than state-of-the-art mixed-signal and digital implementation, respectively.

REFERENCES

[1] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. 2018. Unprocessing Images for Learned Raw Denoising. *CoRR* abs/1811.11127 (2018). arXiv:1811.11127

[2] Weidong Cao, Xin He, Ayan Chakrabarti, and Xuan Zhang. 2019. NeuADC: Neural Network-Inspired RRAM-Based Synthesizable Analog-to-Digital Conversion with Reconfigurable Quantization Support. In *DATE'19*. 1477–1482.

[3] Weidong Cao, Xin He, Ayan Chakrabarti, and Xuan Zhang. 2020. NeuADC: Neural Network-Inspired Synthesizable Analog-to-Digital Conversion. *IEEE TCAD* 39, 9 (2020), 1841–1854.

[4] Weidong Cao, Liu Ke, Ayan Chakrabarti, and Xuan Zhang. 2019. Neural Network-Inspired Analog-to-Digital Conversion to Achieve Super-Resolution with Low-Precision RRAM Devices. In *IEEE/ACM ICCAD'19*. 1–7.

[5] Weidong Cao, Yilong Zhao, Adith Boloor, Yinhe Han, Xuan Zhang, and Li Jiang. 2021. Neural-PIM: Efficient Processing-In-Memory with Neural Approximation of Peripherals. *IEEE Trans. Comput.* (2021), 1–1.

[6] Jaehyuk Choi, Seokjun Park, Jihyun Cho, and Euisik Yoon. 2014. A 3.4-μ W Object-Adaptive CMOS Image Sensor With Embedded Feature Extraction Algorithm for Motion-Triggered Object-of-Interest Imaging. *IEEE JSSC* 49, 1 (2014), 289–300.

[7] N. Dalal and B. Triggs. 2005. Histograms of oriented gradients for human detection. In *IEEE CVPR'05*, Vol. 1. 886–893 vol. 1.

[8] B. Mudassar et al. 2019. CAMEL: An Adaptive Camera With Embedded Machine Learning-Based Sensor Parameter Control. *IEEE JETCAS* 9, 3 (2019), 498–508.

[9] H. Kim et al. 2016. A Delta-Readout Scheme for Low-Power CMOS Image Sensors With Multi-Column-Parallel SAR ADCs. *IEEE JSSC* 51, 10 (2016), 2262–2273.

[10] H. Tsugawa et al. 2017. Pixel/DRAM/logic 3-layer stacked CMOS image sensor technology. In *IEEE IEDM'17*. 3.2.1–3.2.4.

[11] K. Mizuno et al. 2012. Architectural Study of HOG Feature Extraction Processor for Real-Time Object Detection. In *IEEE Workshop on SiPS'12*. 197–202.

[12] R. Eki et al. 2021. 9.6 A 1/2.3inch 12.3Mpixel with On-Chip 4.97TOPS/W CNN Processor Back-Illuminated Stacked CMOS Image Sensor. In *IEEE ISSCC'21*, Vol. 64. 154–156.

[13] Kurt Hornik. 1991. Approximation Capabilities of Multilayer Feedforward Networks. *Neural Netw.* 4, 2 (March 1991), 251–257.

[14] WooYeon Jeong and Kyoung Mu Lee. 2005. CV-SLAM: a new ceiling vision-based SLAM technique. In *IEEE/RSJ IROS'05*. 3195–3200.

[15] Vadim Kantorov and Ivan Laptev. 2014. Efficient Feature Extraction, Encoding, and Classification for Action Recognition. In *IEEE CVPR'14*. 2593–2600.

[16] Guruprasad Katti, Michele Stucchi, Kristin De Meyer, and Wim Dehaene. 2010. Electrical Modeling and Characterization of Through Silicon via for Three-Dimensional ICs. *IEEE TED* 57, 1 (2010), 256–262.

[17] Boxun Li, Peng Gu, Yi Shan, Yu Wang, Yiran Chen, and Huazhong Yang. 2015. RRAM-Based Analog Approximate Computing. *IEEE TCAD* 34, 12 (2015), 1905–1917.

[18] Robert LiKamWa, Yunhui Hou, Yuan Gao, Mia Polansky, and Lin Zhong. 2016. RedEye: Analog ConvNet Image Sensor Architecture for Continuous Mobile Vision. In *ACM/IEEE ISCA'16*. 255–266.

[19] Zheyu Liu, Erxiang Ren, Fei Qiao, Qi Wei, Xinjun Liu, Li Luo, Huichan Zhao, and Huazhong Yang. 2020. NS-CIM: A Current-Mode Computation-in-Memory Architecture Enabling Near-Sensor Processing for Intelligent IoT Vision Nodes. *IEEE TCAS-I* 67, 9 (2020), 2909–2922.

[20] Yandong Luo, Xu Han, Zhilu Ye, Hugh Barnaby, Jae-Sun Seo, and Shimeng Yu. 2020. Array-Level Programming of 3-Bit per Cell Resistive Memory and Its Application for Deep Neural Network Inference. *IEEE TED* 67, 11 (2020), 4621–4625.

[21] Makoto Motoyoshi. 2009. Through-Silicon Via (TSV). *Proc. IEEE* 97, 1 (2009), 43–48.

[22] S. Munder and D.M. Gavrila. 2006. An Experimental Study on Pedestrian Classification. *IEEE TPAMI* 28, 11 (2006), 1863–1868.

[23] M. O'Halloran and R. Sarpeshkar. 2004. A 10-nW 12-bit accurate analog storage cell with 10-aA leakage. *IEEE JSSC* 39, 11 (2004), 1985–1996.

[24] Matt Poremba, Sparsh Mittal, Dong Li, Jeffrey S. Vetter, and Yuan Xie. 2015. DESTINY: A tool for modeling emerging 3D NVM and eDRAM caches. In *DATE'15*. 1543–1546.

[25] Amr Suleiman, Yu-Hsin Chen, Joel Emer, and Vivienne Sze. 2017. Towards closing the energy gap between HOG and CNN features for embedded vision. In *IEEE ISCAS'17*. 1–4.

[26] Amr Suleiman, Zhengdong Zhang, and Vivienne Sze. 2017. A 58.6 mW 30 Frames/s Real-Time Programmable Multiobject Detection Accelerator With Deformable Parts Models on Full HD 1920 × 1080 Videos. *IEEE JSSC* 52, 3 (2017), 844–855.

[27] Skyler Weaver, Benjamin Hershberg, and Un-Ku Moon. 2014. Digitally Synthesized Stochastic Flash ADC Using Only Standard Digital Cells. *IEEE TCAS-I* 61, 1 (2014), 84–91.

[28] Christopher Young, Alex Omid-Zohoor, Pedram Lajevardi, and Boris Murmann. 2019. A Data-Compressive 1.5/2.75-bit Log-Gradient QVGA Image Sensor With Multi-Scale Readout for Always-On Object Detection. *IEEE JSSC* 54, 11 (2019), 2932–2946.