Energy-Efficient HOG-based Object Detection at 1080HD 60 fps with Multi-Scale Support

Energy-Efficient HOG-based Object Detection at 1080HD 60 fps with Multi-Scale Support

Amr Suleiman, Vivienne Sze
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
Email: suleiman, sze@mit.edu

***Abstract*—In this paper, we present a real-time and energy-efficient multi-scale object detector using Histogram of Oriented Gradient (HOG) features and Support Vector Machine (SVM) classification. Parallel detectors with balanced workload are used to enable processing of multiple scales and increase the throughput such that voltage scaling can be applied to reduce energy consumption. Image pre-processing is also introduced to further reduce power and area cost of the image scales generation. This design can operate on high definition 1080HD video at 60 fps in real-time with a clock rate of 270 MHz, and consumes 45.3 mW (0.36 nJ/pixel) based on post-layout simulations. The ASIC has an area of 490 kgates and 0.538 Mbit on-chip memory in a 45nm SOI CMOS process.**

I. Introduction

Object detection is needed for many embedded vision applications including surveillance, advanced driver assistance systems (ADAS), portable electronics and robotics. For these applications, it is desirable for object detection to be real-time, robust and energy-efficient. Histogram of Oriented Gradients (HOG) is a widely accepted feature for object detection [3], which provides a reasonable trade-off between detection accuracy and complexity compared to alternative richer features [4].

Real-time processing is necessary for applications such as ADAS, and autonomous control in unmanned aircraft vehicles (UAV), where the vehicle needs to react quickly to changing environments. High frame rate enables *faster* detection to allow more time for course correction. High resolution images enable *early* detection by having enough pixels to identify objects at far distances. Finally, in both UAV and portable electronics, the available energy is limited by the battery whose weight and size must be kept to a minimum. For ADAS on the other hand, the power consumption is limited by the heat dissipation [7]. Thus, energy-efficient object detection is also desirable.

Objects can appear in different sizes and distances as shown in Fig. 1. Thus, it is essential that detectors support multiple image scales to detect objects with variable sizes. In this paper, we will describe a hardware-friendly real-time energy-efficient HOG-based object detector, with multi-scale support for robust and accurate detection on high definition video.

II. Previous Work

The majority of published implementations for HOG-based object detection are on CPU and GPU platforms. In addition to consuming power in the hundreds of Watts (e.g., Nvidia 8800 GTX GPU consumes 185W [8]), which is not suitable for embedded applications, they often cannot reach high definition (HD) resolutions. Implementation in [2] achieves high throughput on a GPU at 100 fps using the approach presented in [4] to speed up feature extraction, but with a resolution of 640×480 pixels.

Fig. 1: Images from INRIA person database [1] with different pedestrian sizes based on their distance from the camera.

For higher throughput, FPGA-based implementations have recently been reported. In [6], HOG-based detector is implemented on an FPGA and can process 1080HD (1920×1080 pixels) at 30 fps with a *single* scale support. An ASIC version of this design is presented in [9] with dual cores to enable voltage scaling down to 0.7 V with power consumption of 40.3mW for 1080HD at 30 fps. In [5], the detector is implemented on an FPGA and can process 1080HD at 64 fps multi-scale support by time-multiplexing 18 scales across 3 successive frames. It should be noted that these implementations have relatively large on-chip memory requirements (e.g., [9] uses 1.22 Mbit on ASIC, and [5] uses 7 Mbit on FPGA), which contributes to increased hardware cost.

III. Overview of HOG algorithm

Fig. 2 shows a block diagram of the steps involved in object detection using HOG features. The image is divided into non-overlapping 8×8 pixels patches called *cells*, where gradients are calculated for each pixel. A histogram of the gradient orientations with 9 bins is generated for each cell. The histogram is then normalized by its neighboring cells to increase robustness to texture and illumination variation. For each cell, the normalization is performed across *blocks* of 2×2 cells resulting in a final 36-dimension feature.

As mentioned in Section I, scaling is an important factor in object detection algorithms since there is no prior knowledge about object distance/size. The precision-recall curve shown in Fig. 3 is one way to measure the detection accuracy. Increasing the number of scales, by reducing the scale factor, increases Average Precision (AP)[1] from 0.166 with single scale to 0.4 with scale factor of 1.05 (44 scales per 1080HD frame).

[1]Average precision measures the area under precision-recall curve. Higher average precision means better detection accuracy.

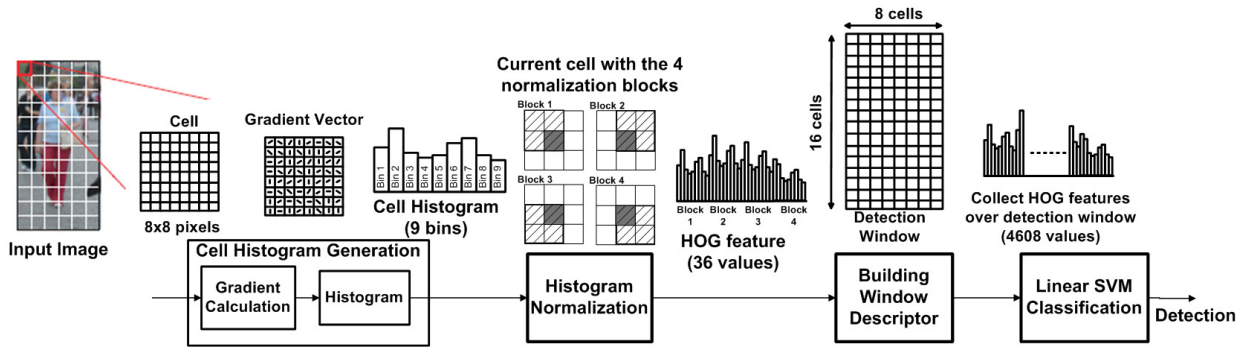

Fig. 2: Object detection algorithm using HOG features.

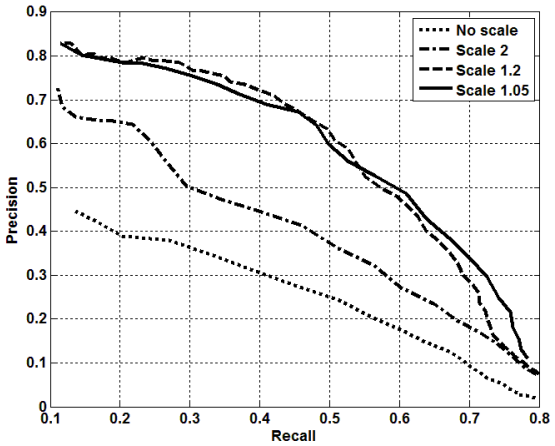

Fig. 3: Precision-Recall curves for pedestrian detection using different scaling factors: single scale (AP=0.166), scale of 2 (AP=0.275), scale of 1.2 (AP=0.391), and scale of 1.05 (AP=0.401, used in original HOG algorithm [3]).

Multi-scale detection is done by training one SVM classifier that has the size of the detection window. An image pyramid is generated, which is composed of multiple scaled versions of each frame. All image pyramid levels are then processed by the same SVM classifier.

IV. Hardware Architecture

A. Detector

The detector can be divided into HOG feature extraction and SVM classification. Feature extraction (Fig. 4) includes cell histogram generation, the histogram buffer, and histogram normalization. SVM classification (Fig. 5) includes multiply-and-accumulate (MAC) units to compute the dot product, the accumulation buffer, and the SVM weights memory.

1) HOG feature extraction: A gradient filter [-1 0 1] is used to generate a pair of horizontal and vertical gradients for each pixel. The orientation and the magnitude of the gradient are then calculated from this pair, and a histogram of 9 bins is generated for the cell. Since the orientation is only used to choose the histogram bin, the actual angle value of the gradient orientation does not need to be calculated. Each gradient bin can be determined by comparing vertical and horizontal gradients multiplied by constant angle tangents representing bins edges. Computing the L2-norm magnitude of gradients requires a square root operation, which is relatively complex for hardware implementations. In this work, an L1-norm is used for the magnitude to avoid using a square root with no impact to detection accuracy.

As shown in Fig. 2, each cell requires its neighboring 8 cells for the normalization process. Accordingly, the resulting 9-bin cell histogram must be stored in a column buffer (0.055 Mbit for a single-scale detector), so that it can be accessed later to compute the normalized histogram. The normalization is done by dividing the 9-bin histogram by the block energy (L2-norm) of each of the four neighbouring blocks. Unlike the gradient magnitude calculation, using L1-norm for normalization results in detection degradation [3]. The square root module is implemented using a non-restoring architecture and is shared across the four blocks. Finally, 9 sequential fixed point dividers are used to generate the final HOG feature, which is a 36-dimension vector for each cell.

2) SVM Classification: Linear SVM classifiers are usually used for object detection in conjunction with HOG features [3]. In this work, the classifier is trained off-line and the SVM weights are stored in an on-chip SRAM, so that the detector can be configured for different objects. The bit-width of the SVM weights is reduced to minimize both the memory size and bandwidth. The 4608 SVM weights are quantized to a 4-bit signed fixed-point representation, with a total memory size of 0.018 Mbit. HOG feature bit-width is chosen to be 9-bit signed fixed-point representation to maintain the detection accuracy.

The HOG feature of each cell is immediately used for classification once it is extracted so that it is never buffered or recomputed. All calculations that include the HOG feature must be completed before it is thrown away. Accordingly, the SVM classification, which involves a dot product between HOG features and SVM weights, is done using an on-the-fly approach similar to [9].

The on-the-fly classification block is shown in Fig. 5. The processing is done using multiple MAC units. Each MAC contains two multipliers and two adders to compute and accumulate the partial dot product of two values of the window descriptor and the SVM weights. For a 128×64 pixels pedestrian detection window size, each cell is shared with 16×8 windows, but at different positions within each

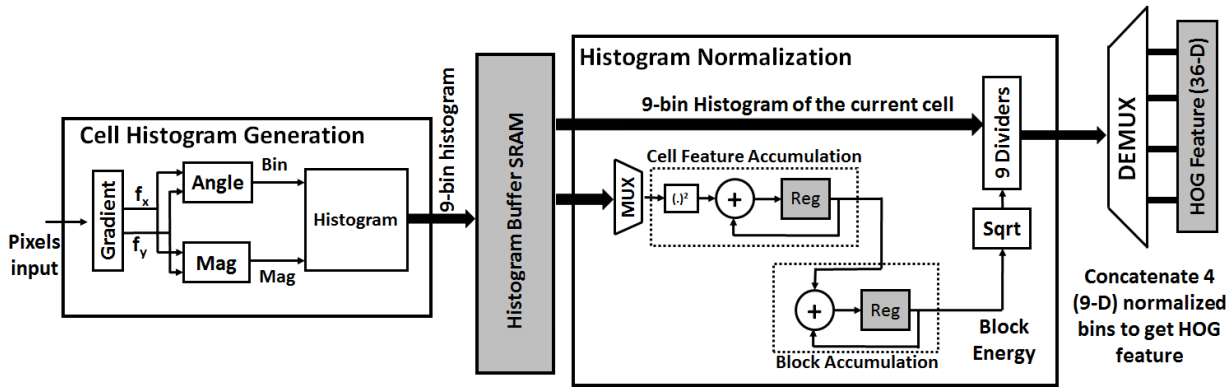

Fig. 4: Block diagram of HOG feature extraction.

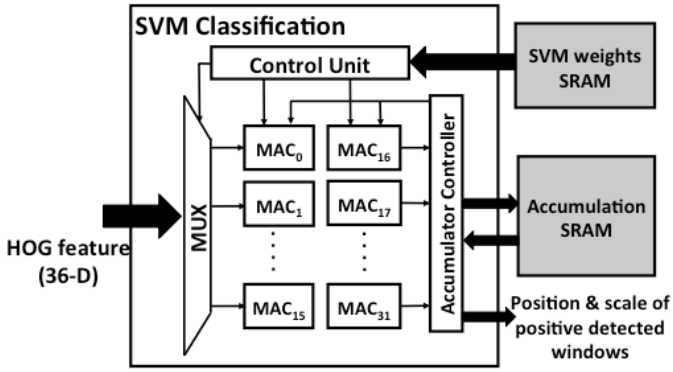

Fig. 5: Block diagram of on-the-fly SVM classification unit.

window. Using this approach, the accumulation values (17-bit) are stored in column buffers instead of the HOG features (36×9-bits) for a 19× reduction in memory (from 0.336 Mbit to 0.018 Mbit) for a 1080HD frame.

B. Scale Generator

1) Scale Factor Selection: An image pyramid is generated for each frame in order to support multi-scale detection. The ratio between the size of successive levels in each dimension is called the scale factor. There is a trade-off in selecting the scale factor between the detection accuracy and number of cells to process. Table I shows an exponential increase in the number of cells per frame as more scales are used (i.e., reducing the scale factor). Using a scale factor of 1.05 in [3] increases the workload by more than 10×. In this work, a scale factor of 1.2 is chosen as it introduces only 1% reduction in AP, with an increase of 3.2× in the workload rather than 10×.

2) Scale Generation Architecture: For a scale factor of 1.2, a total of 12 pyramid levels are generated for a 1080HD frame. Fig. 6 shows a block diagram of the scale generator module. On-the-fly processing is used to generate the scales in order to minimize on-chip memory size.

Pixels are only read once from the external memory. A 3×3 averaging filter is used to low pass filter the original image before downsampling to prevent aliasing, and then pixels are stored in the pixel buffers. The 12 scales are generated

TABLE I: Scale factor effect on detection accuracy, with number of cells per 1080HD frame in the image pyramid.

Scale Factor	AP	#scales	#cells	Increase
Single-scale	0.166	1	32,400	1.0×
2	0.275	4	43,030	1.3×
1.4	0.337	7	65,530	2.0×
1.3	0.372	9	78,660	2.4×
1.2	0.391	12	104,740	3.2×
1.1	0.398	23	184,450	5.7×
1.05	0.401	44	344,220	10.6×

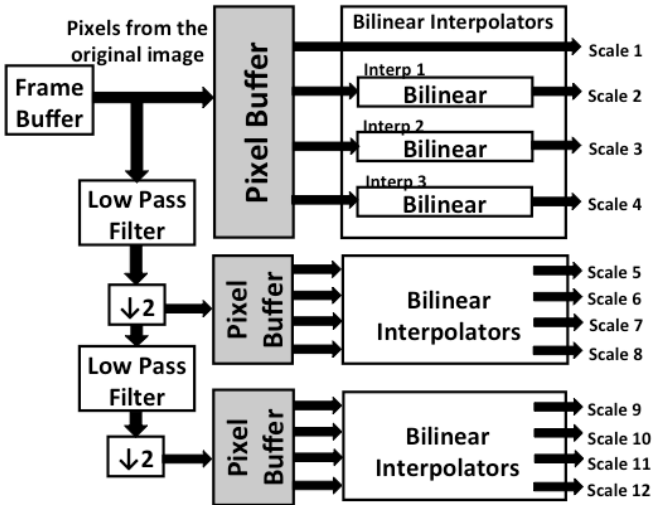

Fig. 6: Scale generation architecture.

as follows: the fifth and ninth levels are approximated with octaves (downsampled by 2) to reduce the number of levels that require interpolation, and three scales are generated from each octave (including the original image) with a scale factor of 1.2 using bilinear interpolation. Three SRAMs are used as pixel buffers for each octave, as illustrated in Fig. 6, and are read by the interpolation block. Interpolation for scaled image generation begins as soon as the required pixels are available in the pixel buffer. The width of the pixel buffers is chosen to cover the required pixel support for the three successive scales.

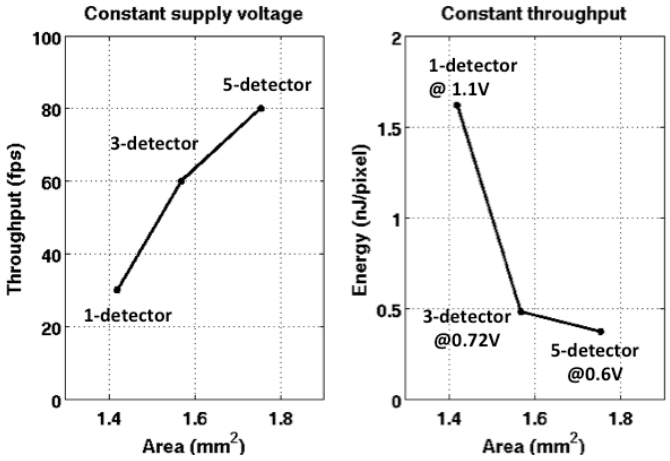

Fig. 7: Throughput with constant supply voltage of 0.72 V (left) and energy with constant throughput of 1080HD at 60 fps (right) for the three different detectors configurations.[2]

C. System Architecture

Multi-scale detection increases the power consumption relative to single-scale due to the scale generation overhead and the processing for the additional scales. If a single detector is used, the frequency and voltage must be increased to process these additional scales while maintaining the throughput of single-scale; thus in this work, multiple detectors are used in order to either maintain or reduce the clock frequency and voltage.

Three different configurations are tested for the shared parallel detectors architecture: one, three and five detectors. In each configuration, all detectors are identical, except that the size of the histogram and SVM accumulator buffers increase based on the number and size of scales processed by each detector. For the parallel configurations, the scales are distributed between the detectors to have a balanced workload. The detectors are also synchronized such that they use the same SVM weights at the same time; thus a single SVM memory can support all detectors with no additional memory bandwidth.

Fig. 7 shows the trade-off between area, power and throughput in each architecture. As expected, large energy reduction is achieved by using three parallel detectors compared to a single detector. Although the five-detector architecture gives a lower energy point, the three-detector design is selected because it offers a better energy versus area trade-off. Fig. 8 shows the overall detection system using three parallel detectors with balanced workload.

V. Image Pre-processing

A. Coarse Resolution of Pixel Intensity

The pixel buffer size is 0.363 Mbit, which is half of the total memory size in the system. The image pixels can be quantized below the original 8-bit to reduce the size of the pixel buffer and to reduce the size of the multipliers in the interpolation block. Fig. 11 shows the AP versus pixel intensity bit-width. As the HOG features are fairly robust to quantization noise, the pixels can be quantized to 4-bit for an AP of 0.372 versus 0.389 for 8-bit.

[2]Energy numbers for 0.6 V and 1.1 V supplies are estimated from a ring oscillator voltage versus power and frequency curves. SRAM minimum voltage is 0.72 V.

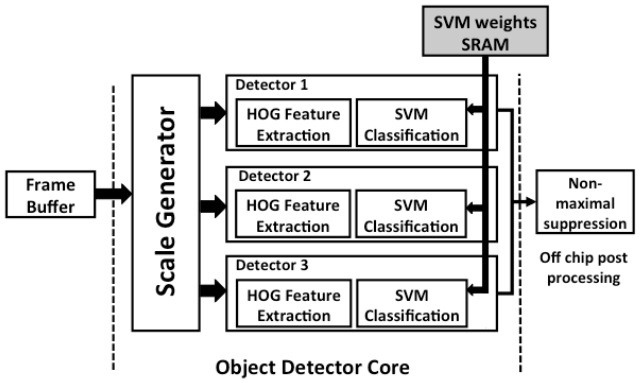

Fig. 8: Object detection system architecture.

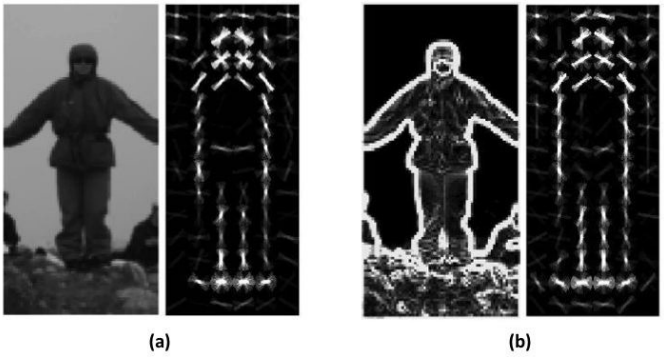

Fig. 9: Pedestrian image in (a) original and (b) gradient representations. Image is taken from INRIA person dataset [1].

B. Detection on Gradient Image

HOG feature is a function of edge orientations and it should have consistent performance for detection on other image representations that preserve edge orientations. Fig. 9 shows two representations of the same pedestrian: one is the original intensity image, and the other is the gradient magnitude image. The gradients are calculated using a simple [-1 0 1] filter. Edges that compose the pedestrian contour are visible in both images. To further demonstrate that detection on gradient images can be reasonable, Fig. 9 shows also a visualization of trained SVM templates for both original and gradient images. Both templates capture similar pedestrian characteristics (e.g. head, shoulders, legs).

The motivation behind processing gradient images is to further reduce the pixel bit-width, and to reduce the switching activity in the design. Fig. 10 shows the histograms for the original and gradient representations of an example image. The original image pixels values are well distributed across the whole dynamic range of 8-bit. However in the gradient image, most of the pixels are concentrated around lower values and do not cover the whole dynamic range. Thus, the gradient image can save one bit in pixel bit-width because of the dynamic range reduction.

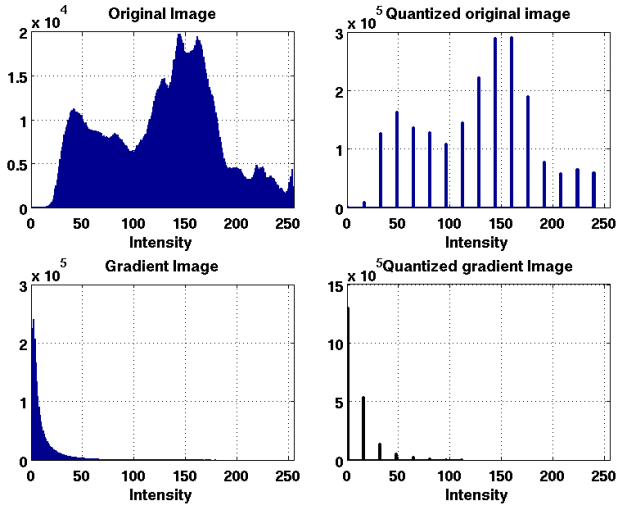

Fig. 10: Histogram of pixel intensities. Left column shows 8-bit bit-width. Right column shows coarse quantization.

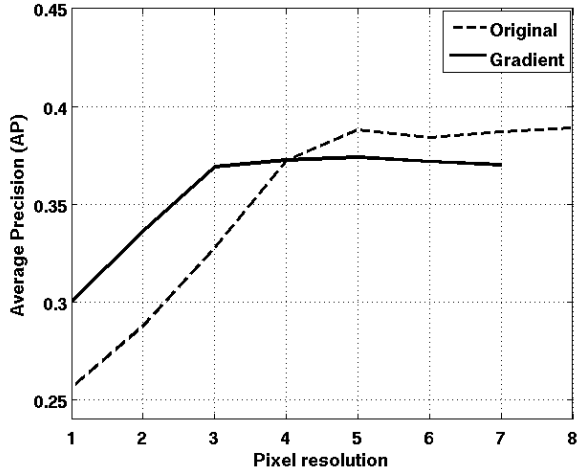

Fig. 11: Average precision (AP) values versus pixel resolution for both original and gradient images.

Fig. 11 shows that original images give better AP at high resolutions, however the difference is less than 2%. At 4-bit pixel bit-width, both gradient and original images have same AP. Reducing the pixel bit-width to 3-bit in the gradient images reduces the AP by only 0.7%, but gives a 25% reduction in pixel buffer size.

VI. Simulation Results

A. Detection Accuracy

The INRIA person dataset [1] was used to evaluate the impact of the modified parameters on the detection accuracy. These modifications include: using L1-norm for gradient magnitude, fixed-point numbers representation, and generating image pyramid with a scale factor of 1.2. Our implementation, which supports multi-scale detection, without pre-processing is close to the original HOG algorithm [3] with 0.389 AP compared to 0.4. With pre-processing, our implementation gives 0.369 AP.

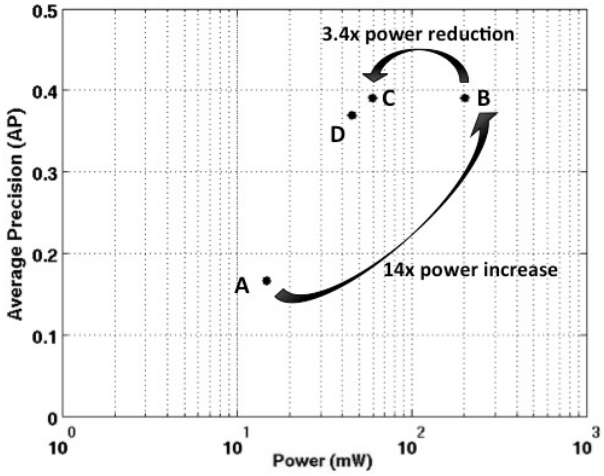

Fig. 12: (A) Single-scale with one detector at 0.6 V. (B) Multi-scale with one detector at 1.1 V. (C) Multi-scale with three detectors at 0.72 V. (D) Multi-scale with three detectors and pre-processing at 0.72 V.[3]

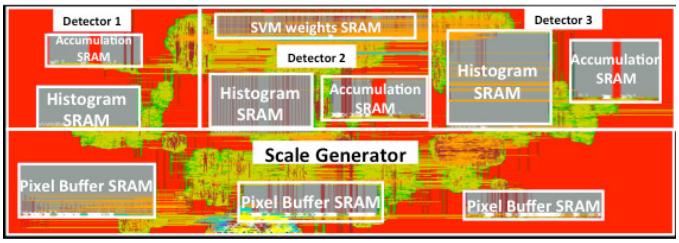

Fig. 13: Layout of the object detector core.

B. Architectural and algorithmic optimization results

Fig. 12 shows the design space of detection accuracy and power numbers for different architectures at the same throughput (1080HD at 60 fps). As expected, introducing multi-scale detection results in a much higher detection accuracy, but with a large overhead in power consumption (14× power increase from A to B in Fig. 12). Using a parallel detector architecture reduces the power by 3.4× due to less workload per detector, which enables voltage and frequency reduction (from B to C Fig. 12), without affecting detection accuracy.

Converting the input image to gradient, and reducing the pixel resolution down to 3-bit, results in a 42% power reduction in the scale generator block. The pixel buffer size is reduced from 0.363 Mbit to 0.137 Mbit. Smaller multipliers are used in the interpolation unit, resulting in 27% saving in logic area. Detector power is also reduced by 17% due to smaller subtractors and accumulators in the histogram generation unit, and due to reduction in switching activity in the data-path. This pre-processing achieves a 24% overall reduction of the system power (from C to D in Fig. 12).

[3]Energy numbers for 0.6 V and 1.1 V supplies are estimated from a ring oscillator voltage versus power and frequency curves. SRAM minimum voltage is 0.72 V.

TABLE II: Area and power breakdown for object detector architecture for both original and gradient images.

		Original	Gradient
Area (kgates)	Pre-processing	n/a	7
	Scale Generator	240	167
	Detector 1	90	86
	Detector 2	102	100
	Detector 3	133	130
	Total	**565**	**490**
Power (mW)	Pre-processing	n/a	0.5
	Scale Generator	17.10	9.30
	Detector 1	13.10	10.10
	Detector 2	11.60	9.50
	Detector 3	10.15	8.05
	SVM memory	7.85	7.85
	Total	**59.80**	**45.30**

TABLE III: Design specifications for three time-shared parallel detectors, with image pre-processing to 3-bit gradient image.

	[9]	This work
Technology	65nm CMOS	45nm SOI CMOS
Area	3.3×1.2 mm^2	2.8×0.96 mm^2
Gate count	502 kgates	490 kgates
Memory Size	1.22 Mbit	0.538 Mbit
Image resolution	1920x1080	1920x1080
Frequency	42.9 MHz	270 MHz
Frame rate	30 fps	60 fps
Scales	1	12
Supply	0.7 V	0.72 V
Power	40.3 mW	45.3 mW
Energy	0.648 nJ/pixel	0.364 nJ/pixel

C. Hardware Complexity

The final architecture of the multi-scale HOG-based object detector was implemented in a 45nm SOI CMOS process. It has three parallel detectors, a scale generator and a pre-processing unit. The core layout is shown in Fig. 13. The area and power breakdown for both 8-bit original image and 3-bit gradient image detectors are shown in Table II. The pre-processing required for the gradient image detection architecture introduces very small area and power overhead. Table III presents a comparison between this work and the ASIC implementation in [9]. This design can process 1080HD video at 60 fps, with total power consumption of 45.3 mW at a supply voltage of 0.72V. Although this work supports multi-scale detection, its energy per pixel is less than the single scale detector reported in [9] and on-chip memory size is only 0.538 Mbit.

REFERENCES

[1] INRIA Person Dataset. http://pascal.inrialpes.fr/data/human/.

[2] R. Benenson et al. Pedestrian detection at 100 frames per second. In *CVPR*, 2012.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[4] P. Dollar et al. The Fastest Pedestrian Detector in the West. In *BMVC*, 2010.

[5] M. Hahnle et al. FPGA-Based Real-Time Pedestrian Detection on High-Resolution Images. In *CVPR*, 2013.

[6] K. Mizuno et al. Architectural Study of HOG Feature Extraction Processor for Real-Time Object Detection. In *SiPS*, 2012.

[7] B. J. Myers, B. and J. Ratell. Embedded Electronics in Electro-Mechanical Systems for Automotive Applications. *SAE Technical Paper*, 2001.

[8] J. Oh et al. Low-Power, Real-Time Object-Recognition Processors for Mobile Vision Systems. *IEEE Micro*, 2012.

[9] K. Takagi et al. A sub-100-milliwatt dual-core HOG accelerator VLSI for real-time multiple object detection. In *ICASSP*, 2013.