A 58.6mW 30fps Real-Time Programmable Multi-Object Detection Accelerator with Deformable Parts Models on Full HD 1920×1080 Videos

Amr Suleiman, *Student Member, IEEE,* Zhengdong Zhang, *Student Member, IEEE,* and Vivienne Sze, *Senior Member, IEEE*

***Abstract*—This paper presents a programmable, energy-efficient and real-time object detection hardware accelerator for low power and high throughput applications using deformable parts models, with 2× higher detection accuracy than traditional rigid body models. Three methods are used to address the high computational complexity of 8 deformable parts detection: classification pruning for 33× fewer part classification, vector quantization for 15× memory size reduction, and feature basis projection for 2× reduction in the cost of each classification. The chip was fabricated in a 65nm CMOS technology, and can process full high definition 1920×1080 videos at 60fps without any off-chip storage. The chip has two programmable classification engines for multi-object detection. At 30fps, the chip consumes only 58.6mW (0.94 nJ/pixel, 1168 GOPS/W). At a higher throughput of 60fps, the classification engines can be time multiplexed to detect even more than two object classes. This proposed accelerator enables object detection to be as energy-efficient as video compression, which is found in most cameras today.**

***Keywords*—Deformable parts, object detection, basis projection, pruning.**

I. Introduction

Object detection is one of the fundamental technologies for intelligent vision applications such as Advanced Driver Assistant Systems (ADAS), autonomous control in Unmanned Aerial Vehicles (UAV), mobile robot vision, surveillance systems and portable devices [1]–[5]. These applications and others require real-time processing and low power consumption. Real-time processing with high frame rates is essential in ADAS and UAVs, where the detection has to be fast enough to allow sufficient time for any course correction. For robustness, multi-scale detection is important to detect objects with different sizes and/or distances from the camera (an object's height is inversely proportional to its distance from the camera [6]). Processing high resolution images, such as full high definition (HD) 1920×1080, is required in order to have enough pixels for early detection of far and small objects.

The high computational complexity of object detection algorithms necessitates high throughput hardware implementations to enable real-time processing with low energy consumption [7]–[14]. In both UAV and portable devices, low energy consumption is important because the available energy is limited by the battery whose weight and size must be kept to a minimum [15]. Additionally, heat dissipation is a crucial factor for ADAS applications [16].

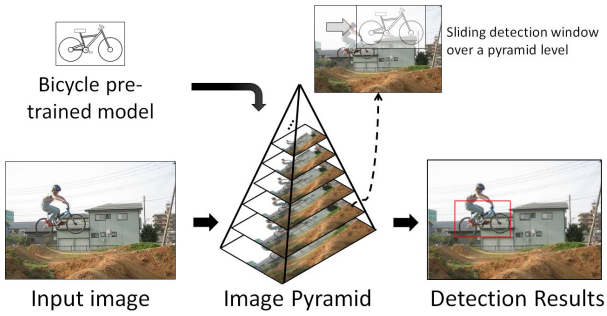


Fig. 1. General object detection approach with a sliding window over an image pyramid for multi-scale detection.

Many image-based object detection algorithms have been developed [17]–[21]. Fig. 1 shows one of the conventional approaches which consists mainly of two processes: localization and classification. In localization, a sliding window scans an image at all positions to find the object. For multi-scale detection, the window scans an image pyramid (multiple down-scaled versions of the image). Multi-scale detection increases the required computation as the image pyramid leads to data expansion, which can be a 100× increase in the number of pixels for a full HD image [10]. In classification, a pre-trained model that captures the characteristics of the target object is used at each sliding window position to label it as a true or a false object.

For robustness, the classification models are trained on features rather than pixels. Histogram of Oriented Gradients (HOG) features, which look at the distribution of edges, are a widely accepted image feature [19]. HOG features provide a favorable trade-off between detection accuracy and complexity compared to alternative richer features [9], [22]. The deformable parts models (DPM) algorithm [20], which is based on HOG features, has demonstrated high detection accuracy. DPM doubles the detection accuracy compared to rigid object detection because of the relatively larger and more flexible models [10]. Lately, convolutional neural networks (CNN) have been widely used in large scale classification systems. Their superior performance comes from their ability to learn

A. Suleiman, Z. Zhang and V. Sze are with the Department of Electrical and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139 USA (email: suleiman@mit.edu).

features with millions of trained parameters [21]. While recent work shows a CNN accelerator that processes 227×227 pixels images in real-time [8], the energy consumption is in the order of hundreds of nJ/pixel. This is much higher than the typical 1 nJ/pixel consumed by multimedia accelerators, such as video decoders [23], in embedded systems.

This work presents an object detection hardware accelerator using the DPM algorithm. The hardware supports multi-scale detection by generating and processing a 12-level image pyramid per frame. Two programmable object classification engines are used to detect two different object classes simultaneously. Three methods are used to address the high computational complexity: classification pruning for 33× fewer classifications, vector quantization for 15× memory size reduction, and feature basis projection for 2× reduction in the cost of each classification. As a result, the proposed accelerator achieves multi-object detection for full HD 1920×1080 videos at 30fps with less than 1 nJ/pixel energy consumption. The accelerator was tested for speeds up to 60fps.

The rest of this paper is organized as follows: Section II describes the DPM algorithm in detail. Then, Section III explains the system architecture of the proposed accelerator with a detailed description of the main building blocks. The chip implementation and evaluation results follow in Section IV. Finally, Section V summarizes the paper.

II. Deformable parts models (DPM)

DPM is a mature and a stable object detection algorithm. While other detection methods are more accurate, DPM and its variants still achieve relatively high detection accuracy in important applications such as pedestrian detection [24]. The main characteristic of this model is its flexibility in detecting deformed objects by having three filters: root, parts and deformation. DPM detection uses HOG features, which are robust against illumination and local shape changes [19]. There are different versions of DPM with different number of parts filters. In this work, we use DPM v5 [25], which detects 8 parts per object. Detection with DPM is described in the following four steps as shown in Fig. 2(a):

- **HOG features generation:** Features are extracted from the input image. For DPM detection, features are extracted at twice the input image resolution, because root and parts filters are processed on different resolutions.
- **Root classification:** The root filter is a pre-trained support vector machine (SVM) [26], which captures the characteristics of the object as a whole as shown in the DPM template in Fig. 2(b). Root classification is the result of a convolution (i.e., dot product) between the root SVM filter and the sliding detection window over the feature pyramid. The output of this process is called *root score*, representing the similarity between the image features and the root filter.
- **Parts classification:** The parts filters are pre-trained SVMs as well, defining several smaller and localized parts of the object as shown in the DPM template in Fig. 2(b). Parts classification is similar to the root classification, but parts convolution is carried out at a pyramid level with twice the resolution of that used for the root filter. This allows detecting fine-grained details of the object. The output of this process is called *parts scores*, representing the similarity between the image features and the parts filters.
- **Deformation:** DPM enables positional variation of the parts relative to their optimum positions (*anchors*). The scores of the parts are penalized with deformation costs defined in the deformation filter as shown in the DPM template in Fig. 2(b). The deformation cost is a function in the part displacement from its anchor (d_x and d_y), where a_1, a_2, b_1 and b_2 are pre-trained parameters.

To summarize, the final DPM score of each detection window is the result of adding the root score and the corresponding 8 maximum deformed parts scores as shown in the following equation:

$$\mathrm{DPM\,score}(x,y) = \mathrm{RS}(x,y) + \sum_{i=1}^{8}\Big(\max_{dx,dy}\big(\mathrm{PS}_i(x,y,dx,dy) - \mathrm{DC}_i(dx,dy)\big)\Big) \quad (1)$$

Where RS, PS and DC stand for root score, part score and deformation cost, respectively.

A. DPM complexity

Although DPM provides higher detection accuracy than rigid body approaches, its high computational cost is a bottleneck for practical applications. Several studies have investigated speeding-up DPM detection. A software-based DPM object detector, described in [27], enables processing 500×500 images at 30fps, but requires a fully loaded Intel Xeon 6-core processor and 32GB of memory. A GPU (Nvidia GeForce GTX TITAN Black GPU with 2,880 cores processes) is used in [28] to accelerate the DPM detection for real-time processing for QVGA 320x240 images; however, the throughput drops for larger resolutions (e.g., 1280x960 at 1.5 fps). These CPUs and GPUs are not suitable for embedded applications because of their high power consumption.

Enabling deformable parts detection doubles the detection accuracy by removing false positives and accurately detecting objects as shown in 2(c). However, this comes at the cost of 35× more computation compared to rigid object detection [10]. This overhead comes from four main factors: 3× larger model size with the parts filters, 4× larger image pyramid size to support parts classification at twice the image resolution relative to the root classification, 1.5× increase due to the deformation computation, and finally 2× increase due to the fact that two DPM models are used (original and flipped version). Our proposed algorithmic and architectural optimizations are carried out to minimize this computational cost while maintaining a reasonable detection accuracy.

III. DPM object detection hardware architecture

A. Overall Architecture

Fig. 3 shows the overall architecture of the proposed DPM object detection hardware accelerator. It consists of three main

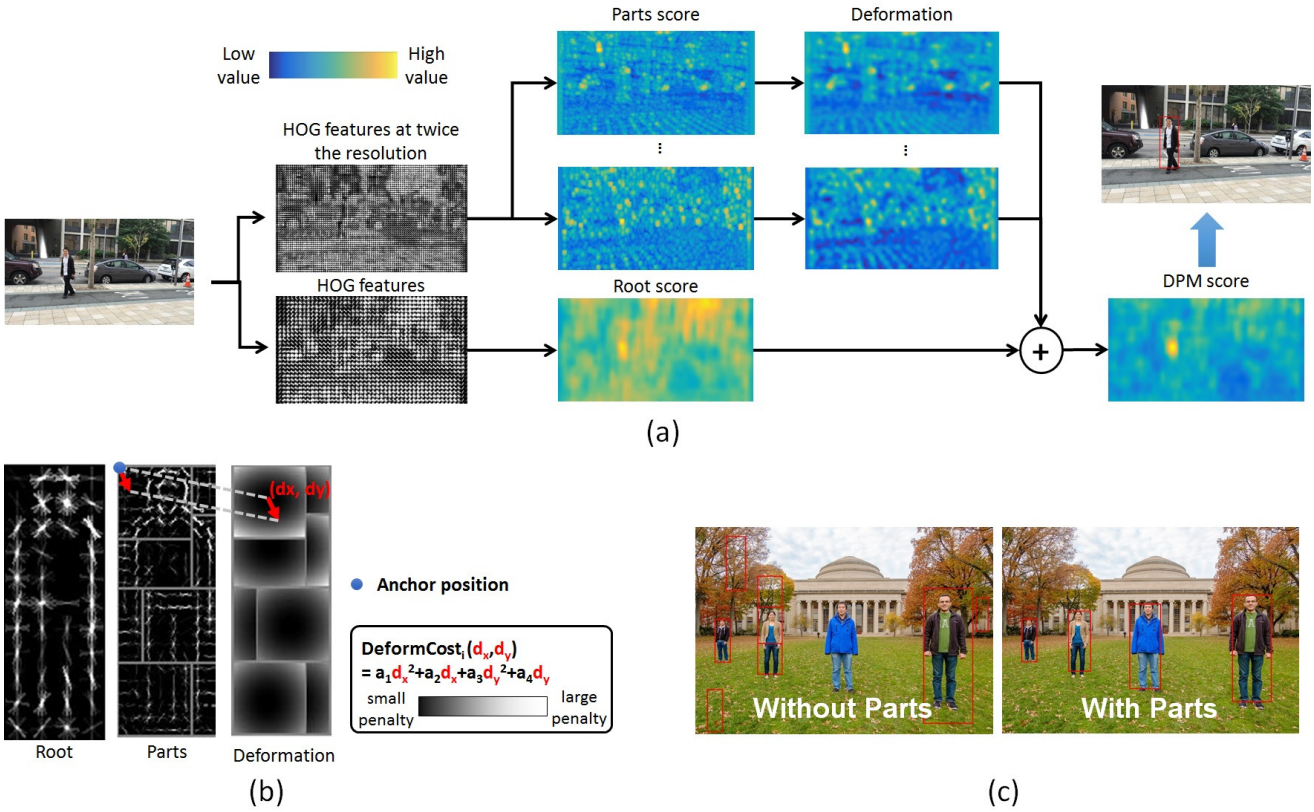


Fig. 2. (a) The pipeline of pedestrian detection with DPM including HOG features generation, root and parts classification and deformation. The figure is inspired by [20]. (b) DPM template for pedestrian detection with root, parts and deformation filters. (c) Pedestrian detection example with and without parts. Parts detection removes false positives and enhances the detection accuracy.

units: feature pyramid generation (FPG), feature storage (FS) and SVM classification engines (CE). All units process data *on-the-fly*, which means all computations are carried out as soon as the data is available such that the input data does not need to be stored. This minimizes the required on-chip memory. The proposed detector hardware accelerator is a standalone unit that takes an input image and outputs the detection results. With the optimizations described in this paper, no external storage is required (e.g., DRAM), which lowers the system level power consumption.

The full HD 1920×1080 input frame is read in a row raster scan order with a throughput of 1 pixel/cycle. The FPG unit processes the pixels and generates a 12-level HOG feature pyramid output (4 octaves, 3 levels/octave). The pyramid size is selected based on favorable trade-off between the computation cost and the detection accuracy. The pyramid contains 87K features vectors, which is 2.7× more vectors compared to a typical full HD frame. The pyramid size, along with the DPM filter height, define the minimum and the maximum object distances from the camera where it can be detected. Fig. 4 shows an example for a specific camera [29] using the proposed architecture.

The feature pyramid is passed to the FS and the two CEs. The FS unit contains line buffers that store HOG features to avoid re-computation (details in Section III-C). The accelerator contains two programmable SVM CEs to support multi-object class detection. The maximum DPM filter size is 128×128 pixels. This large size gives the flexibility to detect different object classes with different aspect ratios. Inside each CE, root classification and 8 parts classification units process the incoming feature pyramid on-the-fly (i.e., no input data needs to be stored on-chip). Finally, deformation unit penalizes the scores of the parts and finds the maximum value. A coarse-to-fine search approach is used for 2.2× speedup in finding the maximum deformed parts score in a 5×5 search window. Each SVM CE outputs the location and the size of the detection windows with DPM scores larger than a programmable threshold. In the following sections, a detailed description of the main building blocks is presented.

B. Feature Pyramid Generation (FPG)

HOG features are calculated by dividing the image into non-overlapping 8×8 pixels patches called *cells*. A histogram of edge orientations is calculated for each cell, and normalized with respect to the surrounding cells [19]. Fig. 5(a) shows the conventional approach to generate the HOG feature pyramid, where the image pyramid is generated first from the input image using bilinear interpolation, and then histograms of

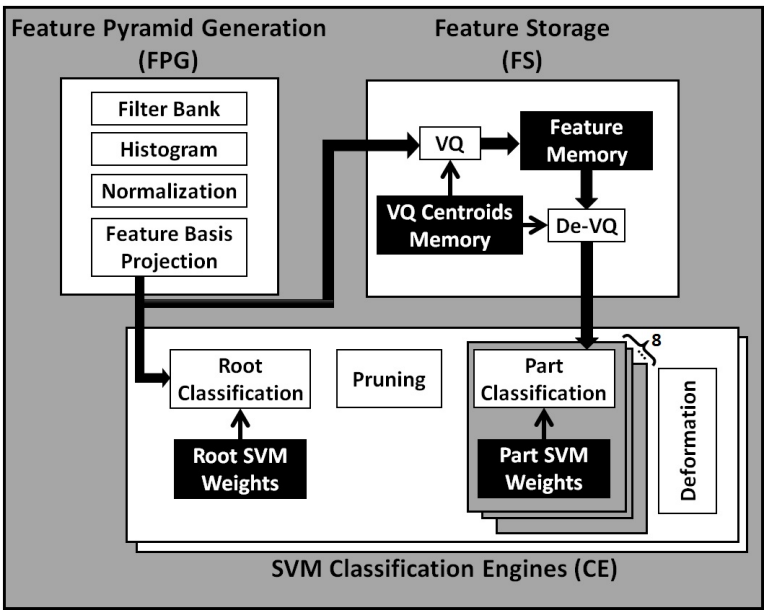


Fig. 3. Overall architecture of the proposed DPM object detection hardware accelerator.

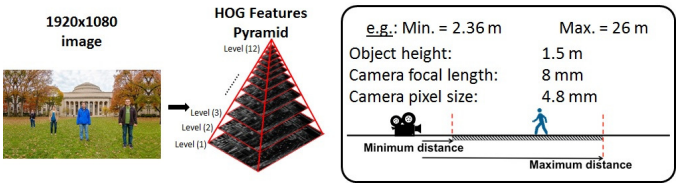


Fig. 4. An example of the maximum and the minimum detectable distances for pedestrian detection with a specific camera parameters [29] using the proposed architecture.

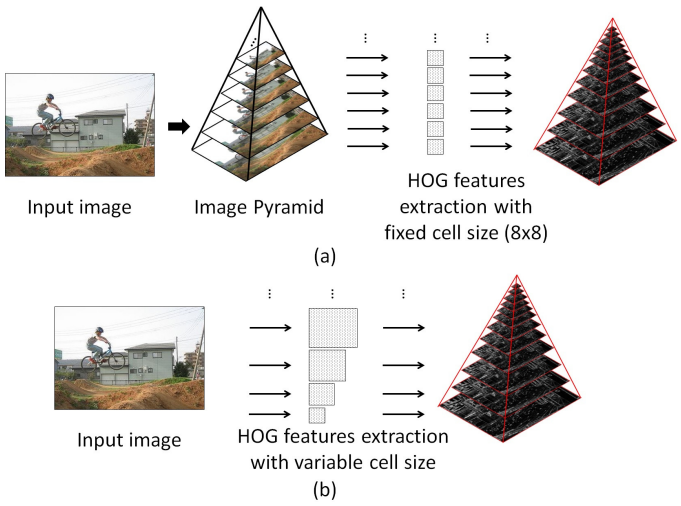


Fig. 5. Generating HOG feature pyramid by (a) generating image pyramid first then calculating 8×8 pixels cells histograms, or (b) calculating histograms of different cell sizes corresponding to each pyramid level.

fixed size cells are generated [9]. In this work, the image pyramid is skipped to eliminate the bilinear interpolation, and histograms of different cell sizes are generated directly from the input image as shown in Fig. 5(b). The 12 cell sizes are 8×8, 10×10, 13×13, 16×16, 20×20, 26×26, 32×32, 40×40, 52×52, 64×64, 80×80 and 104×104. The ratio between each cell size and the original 8×8 pixels cell defines the scaling factor of the corresponding pyramid level.

The FPG unit architecture is shown in Fig. 6. Each input pixel is processed by 12 parallel low pass and gradient filters, which calculate edge angles and magnitudes. The low pass filters prevent aliasing by removing high frequencies in the input image. The cut-off frequency of each filter is proportional to the scaling factor of the corresponding pyramid level. Since the input pixels are coming in row raster scan order, partial histograms are calculated on-the-fly for all the cells that contain the input pixel. The resulting partial histograms of different pyramid levels are available at different times, which is a function of the corresponding cell size. Finally, three parallel engines that compute the histogram and normalization are time-shared to balance the workload and maximize the throughput. HOG features generation is synchronized between different pyramid levels.

The proposed architecture can process input images at 1 pixel/cycle throughput. Eliminating the bilinear interpolation reduces the FPG area by 29.5% and achieves 20.3% power reduction. Additionally, several line buffers are removed but the overall memory size is reduced by only 6.3% because of an increase in the line buffers in the histogram engines.

C. Classification Pruning

With the 87K HOG features generated per full HD frame, on-the-fly root classification is used for minimal on-chip stor-

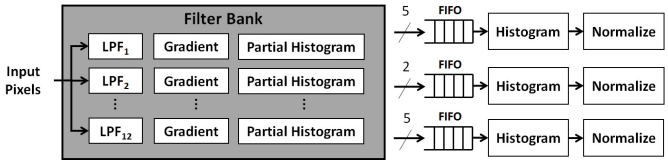


Fig. 6. The FPG unit architecture with multi cell size histogram.

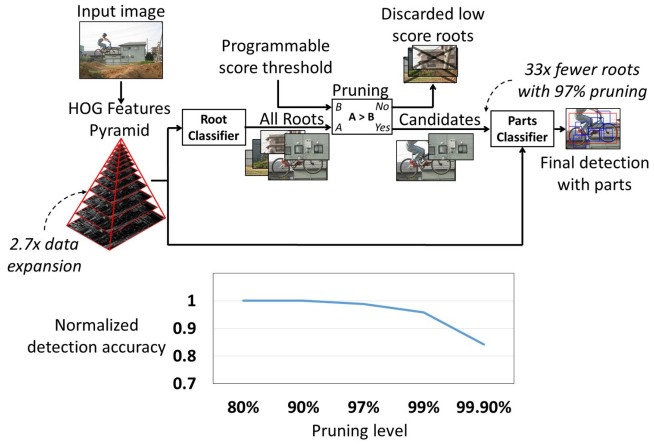


Fig. 7. DPM detection with classification pruning.

age similar to [9], where partial dot products are accumulated in SRAMs. Using the same approach with parts requires large accumulation SRAMs (more than 800KB), which is undesirable. Fortunately, most of the parts classification process can be skipped without affecting the detection accuracy. Specifically, pyramid regions that give very low root scores are almost guaranteed not to generate high DPM scores after adding the deformed parts scores. This is exploited by the accelerator to reduce both computation and memory required for parts classification.

Cascaded classifiers were used in [30] to prune and speed up the classification process; however, a simpler approach is used in the proposed accelerator. Fig. 7 shows the classification pruning algorithm. A pruning threshold is set to discard most of the feature pyramid based on the root scores. Since the root classification is not accurate, the pruning threshold has to be low enough to make sure that true objects are kept along with some false positives. The threshold is programmable for different scenes and objects. Parts classification processes the candidate regions only and filters out the false positives. Increasing the pruning threshold (i.e., discarding more regions of the feature pyramid) reduces the required computation in the proceeding units. The proposed architecture is designed to achieve real-time processing with a minimum pruning level of 80%. Fig. 7 shows that an aggressive 97% pruning can be used without much degradation in the detection accuracy. At this high pruning level, a 33× less parts classification is carried out, which results in a 10× classification power reduction.

With pruning, the parts classification waits for the root scores to decide whether to process a region or not as shown

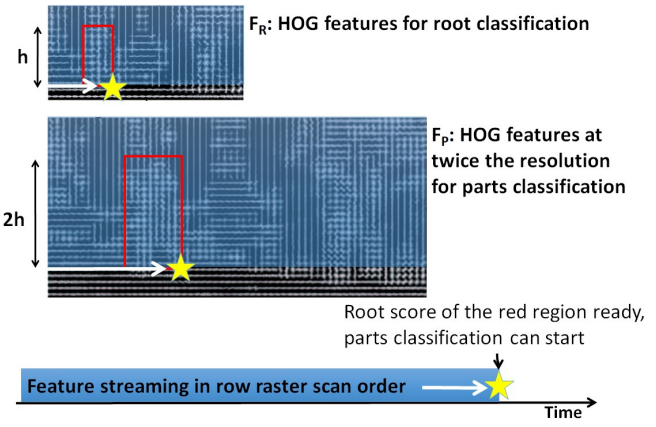


Fig. 8. HOG features streaming example for features re-use support.

in Fig. 7. This means that the root and parts classifications can no longer run in parallel on the same HOG features. As a result, feature storage (FS) is required to enable HOG features re-use and avoid re-computation. Fig. 8 shows an example of the timing of HOG features streaming in a row raster scan order in two pyramid levels (F_R and F_P) that the root and parts classifications are processing. As described in Section III-B, HOG features are always in sync between pyramid levels. After the last HOG feature arrives in the root detection window represented by the red box, the root score will be ready and parts classification of this box can start, assuming that this window is not pruned. The required HOG features for parts classification is shown in the red box in F_P, which is twice the height and twice the width of the red box in F_R. Since the accelerator supports a maximum DPM filter height of 128 pixels (i.e., maximum root detection window height h = 128/8 = 16 cells), the FS line buffers have to store twice this height to support parts classification. Hence, 32 rows of HOG features in all the pyramid levels that the parts classification process are stored in the FS line buffers. A memory size of 572KB is required to store the features for a full HD frame with 12 pyramid levels. To reduce the memory size, vector quantization is used.

D. Vector quantization (VQ)

The VQ technique was used in a DPM implementation on CPU in [31] to speed up the classification by replacing it with lock up tables. In this work, VQ is used instead as an approach to reduce the on-chip memory requirements. Fig. 9 shows a simple visualization of VQ using K-means clustering in 2-D space. The whole space is divided into 8 clusters (L_0 to L_7), each has a centroid (the black X's, C_0 to C_7). VQ is done by representing all the red dots within a cluster by its corresponding centroid. Fig. 9 shows a 2-D example where 64 red dots are represented by 8 centroids after quantization. This means that a 3-bit number, to identify which centroid, can be used rather than a 6-bit number, to identify which dot.

In the proposed architecture, the HOG features space has 13 dimensions. 256 clusters and their corresponding centroids (C_0 to C_{255}) are learned offline from a large number of HOG

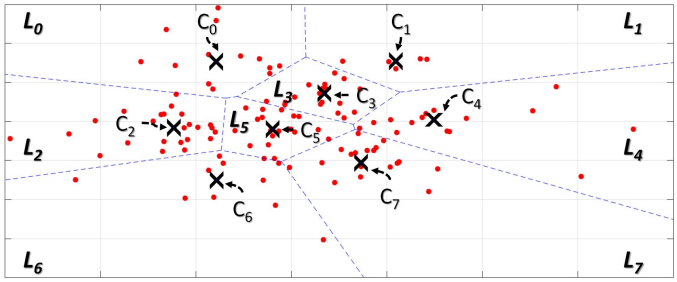


Fig. 9. Visualization of K-means clustering in 2-D space.

features. The selection of the number of clusters is a trade-off between the size of HOG features storage and the quantization error which affects the overall detection accuracy. Fig. 10 shows the architecture of the VQ unit. The input HOG features are compared to the 256 centroids and the nearest one is selected. Parallelism is required to process the 2.6 million HOG features generated per second by the FPG to support full HD video at 30fps. Three VQ engines are used to process three HOG features in parallel (f_1, f_2 and f_3). In each engine, 8 parallel K-means are carried out with each group of 32 centroids separately. Sum of squared difference (SSD) is used to calculate the distances between the centroids and the input HOG features, and the centroid with the minimum distance is selected. A final K-means selects the nearest centroid out of the eight groups using the pre-computed distances, resulting in the output three quantized features (Qf_1, Qf_2 and Qf_3).

The 256 centroids are stored in the VQ centroids memory which is shared between the three VQ engines to reduce bandwidth. In this architecture, with the required bandwidth and size, centroids are stored in a register file which has 22% smaller area and 52% less power consumption compared to SRAM. Power saving comes from the fact that centroids memory can be considered a read-only memory; it is only loaded during start-up. During the VQ process, the centroids are only read from the register file and the registers' clock is always gated. The only switching happening is the address decoding to select a centroid.

A single HOG feature is compressed from 143-bit to 8-bit, which represents one of the 256 centroids. The parallelism in this architecture delivers a 22× increase in throughput at the cost of only 2× increase in the VQ unit area, mainly because of the sharing of the centroids memory. Overall, VQ enables more than 15× memory size and bandwidth reduction in the FS memories from 572KB to 32KB, making it suitable to be implemented on-chip. A 10× power reduction is achieved compared to storing the HOG features off-chip in a DRAM. Quantized features are used in the parts classification only to minimize the detection accuracy losses due to quantization error.

E. Feature Basis Projection

For the supported 12-level HOG feature pyramid, more than 250 million multiplications are carried out per full HD frame in the classification process. To reduce the cost of the classification, the features are projected into a new sparse

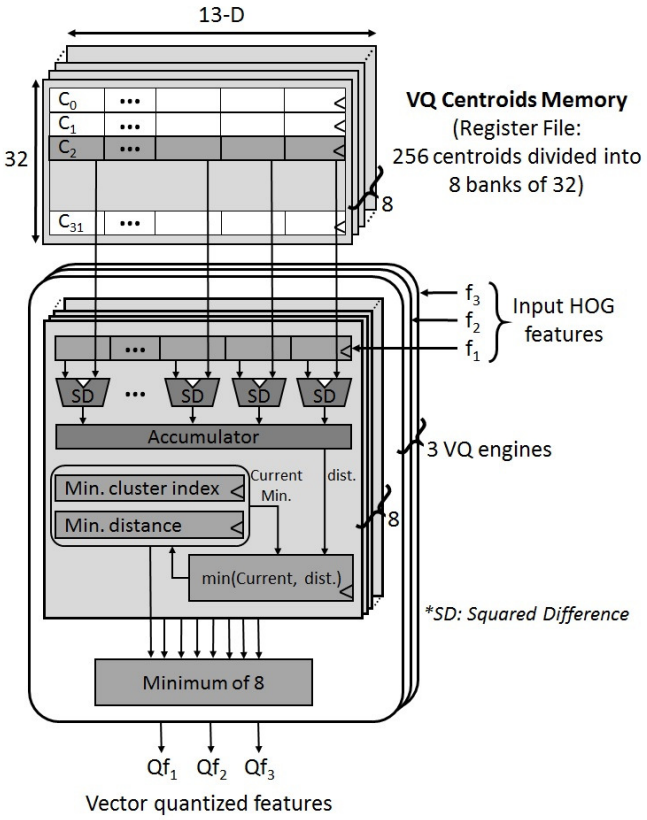


Fig. 10. Vector quantization parallel architecture.

space, hence multiplications with zeros can be skipped. Fig. 11 shows the learning process to design the new sparse space and calculate the corresponding basis vectors (S). Starting from the trained DPM filters weights of different object classes, the 13-D weights per cell are concatenated in a big matrix (W). The optimization problem shown in Fig. 11 is solved to find the basis vectors (S) such that the error between (W) and ($S\alpha$) is minimized, where (α) is the projected sparse weights. The optimization is solved with a constraint on sparsity, where at most 6 out of the 13 new weights per cell are non-zero. This guarantees more than 50% sparsity in the projected filters weights.

Fig. 11 shows also the resulting sparsity after solving the aforementioned optimization problem. The histograms of the SVM filters weights show an increase in the percentage of zeros from 7% to 56% after projection. Mathematically, the classification dot product between HOG features (H) and SVM filters weights (W) is mapped to a new dot product between the projected features (P) and the projected weights (α). Hence, HOG features have to be projected to the new space using the basis vectors (S).

Fig. 12 shows the architecture of the feature basis projection unit along with the root classification in the new sparse space. The projection is carried out only once per HOG features, and the resulting projected features are used by the classification to

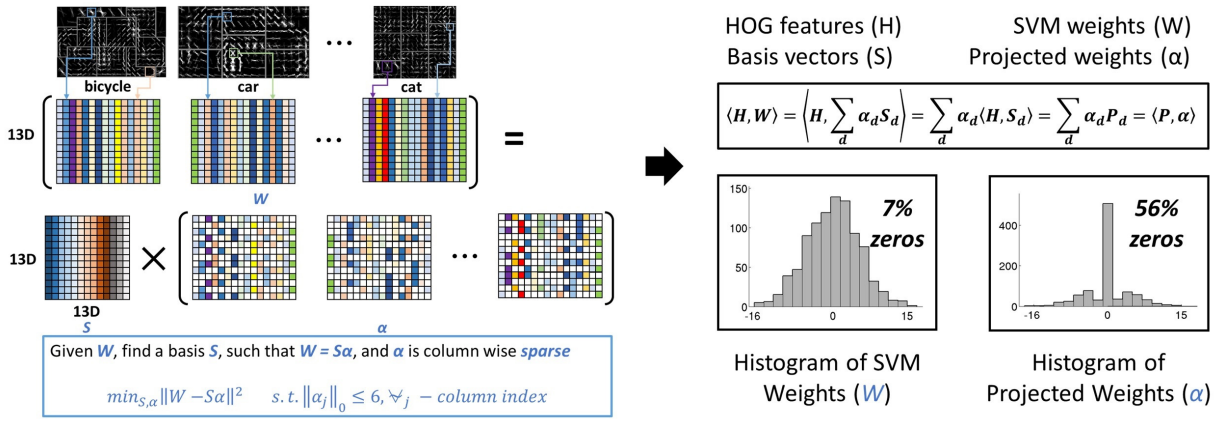


Fig. 11. Learning the feature projection basis and the resulting sparsity.

perform many dot products with all the SVM weights (i.e., an average of 100 dot products per a single HOG feature). A dot product between the input 13-D HOG features (H) and each one of the 13 basis vectors (S_0 to S_{12}) results in the projected features (P), which are 13-D vectors as well.

While the new space is guaranteed to be sparse, the zeros positions in the projected weights (α) with respect to the projected features (P) is not fixed. Accordingly, zero weights are not stored in the SVM weights memory, but a 13-bit flag is stored to indicate the zero positions. The required storage per SVM weights cell is reduced from 65-bit to 43-bit, resulting in a 34% reduction in the SVM weights memory size and bandwidth. The root classification unit consists of 4 parallel engines performing selected MACs. In each engine, a crossbar routes the input 13-D projected features (P) into the multipliers guided by the 13-bit flag. Only 6 multipliers are used because of the guaranteed sparsity designed by the basis projection. Partial scores are accumulated in two steps, a register file and a scores SRAM, to reduce the SRAM bandwidth. The feature basis projection results in only 0.4% area overhead. The classification in the new sparse space reduces number of multiplications by 2× and reduces the overall classification power by 43%.

IV. Implementation and Results

A. Chip Implementation Results

Fig. 13 shows the die photo of the proposed DPM object detection hardware accelerator with area and power breakdowns. FPG takes about one third of the chip area, while the proposed optimizations result in small CE, which each are less than one quarter of the chip. The logic power consumes 57% of the overall chip power while SRAM power consumes 21%. This logic power includes all the registers, some of which are used to replace SRAMs as discussed earlier. Due to these memory registers, the clock tree consumes 20% of the total power.

Table I summarized the chip specifications. The proposed accelerator is implemented in a 65nm CMOS technology with 3.28 million NAND2 equivalent logic gate count and 280.1KB SRAM. The chip has a measured throughput of full

TABLE I. Chip specifications.

Technology	65nm CMOS
Chip size	4.0×4.0 mm^2
Core size	3.58×3.58 mm^2
Gate count	3283 kgates
SRAM	280.1KB
Input resolution	1920×1080
Supply voltage	0.77 - 1.11 V
Frequency	62.5 - 125 MHz
Frame rate	30 - 60 fps
Average power	58.6 - 216.5 mW
Energy	0.94 - 1.74 nJ/pixel
GOPS/W	1168.7 - 623.8
GOPS/mm^2	4.25 - 8.56
DRAM BW (MB/s)	59.6 - 119.2

HD 1920×1080 videos at 30fps at 0.77V while consuming 58.6mW, resulting in a peak performance of 1168 GOPS/W and an energy efficiency of 0.94nJ/pixel. Fig. 14 shows the throughput and energy/pixel versus the supply voltage. The chip can operate at a throughput of 60fps at 1.11V. At 60fps, the two classification engines can be time multiplexed to detect even more than two object classes in real-time. From the system point of view, our optimizations minimize the external DRAM bandwidth (BW) down to only 59.6 MB/s to 119.2 MB/s for a throughput of 30fps to 60fps respectively. This BW is just used to read the input pixel stream once and no other external storage is required throughout the pipeline, which reduces the overall system energy consumption.

Fig. 15 shows the scalability of the proposed architecture at 30fps and 0.77V. The two detectors, and the deformable parts detection for each detector, can be switched on or off for power savings based on the application. This scalability is controlled by four external inputs that directly gate the clock at the unit-level. Although the clock tree gating is left to the synthesis tool, this unit-level logic is manually instantiated

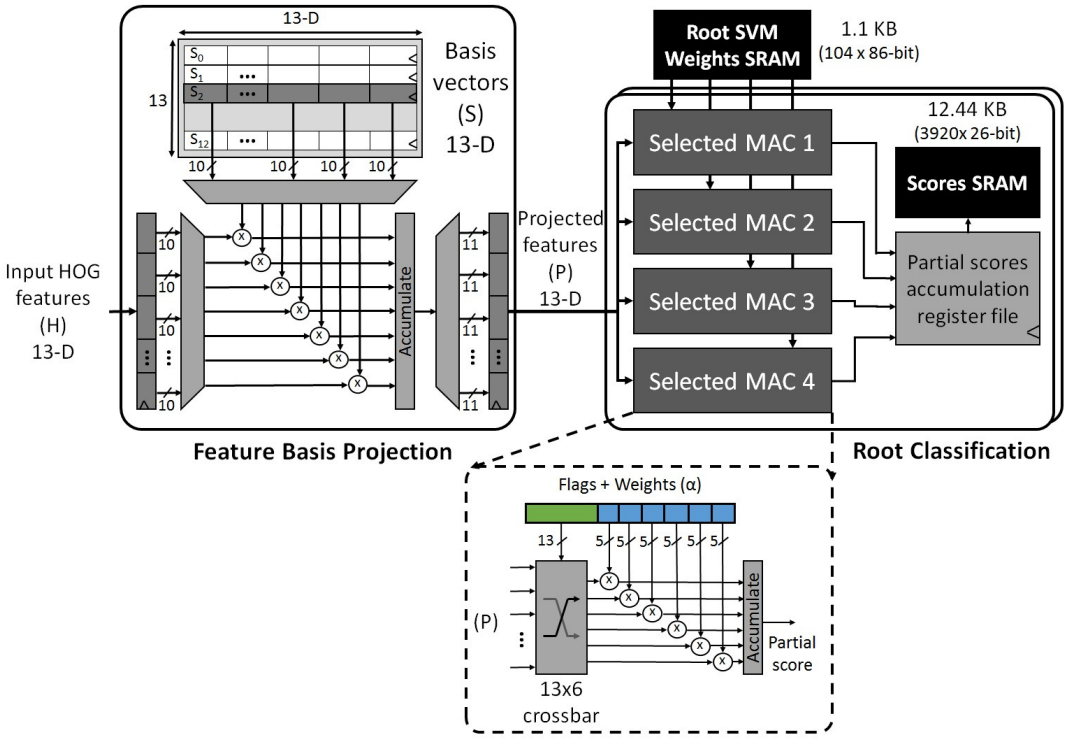


Fig. 12. Feature basis projection architecture and root classification in the new sparse space.

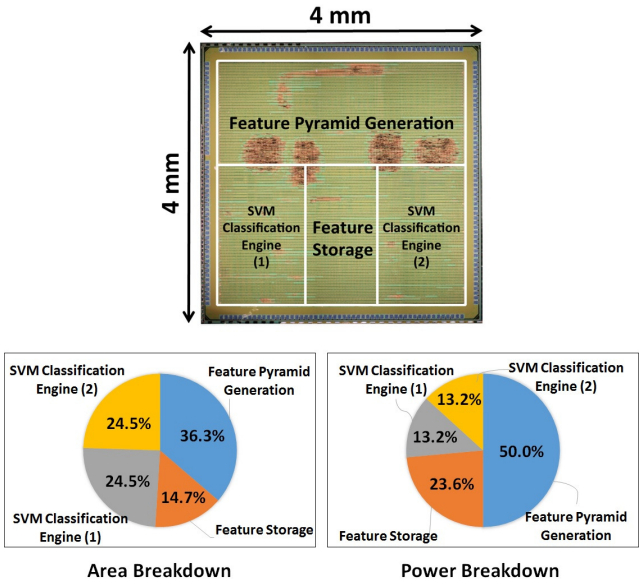


Fig. 13. Chip die photo and breakdowns at 30fps and 0.77V.

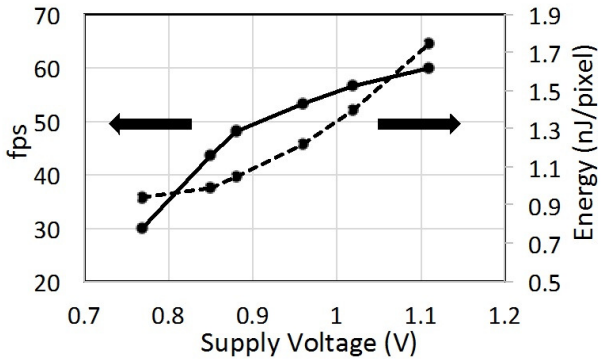


Fig. 14. Energy and throughput versus supply voltage.

during the design to turn off the clock at the input of each unit. The unit-level clock gating ensures that no unnecessary switching is happening inside the clock tree of the units that are not used. Our optimizations result in a lightweight DPM classification (root and parts), which consumes only 15% of the overall power of a single detector. HOG features generation and feature storage are shared between detectors, so enabling the second detector increases the power consumption by only 19%.

B. Evaluation Results

To validate the fabricated chip, a demonstration system is developed for real-time object detection as shown in Fig. 16. It is composed of the chip board, a Xilinx FPGA VC707 development board, a video camera [29] and an LCD display. All the vision processing is done on-chip, and the FPGA board

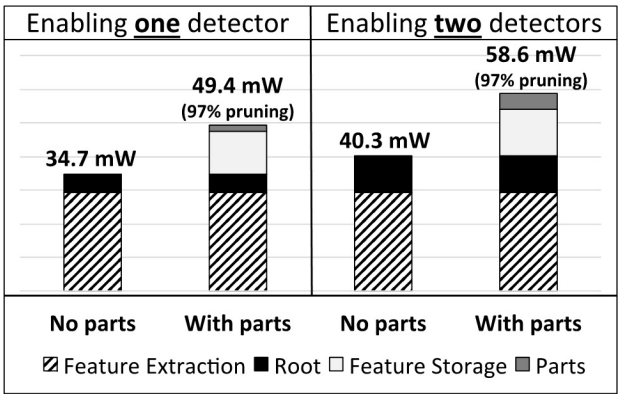


Fig. 15. Chip power scalability at 30fps and 0.77V supply voltage.

is only used for interfacing and configuring the chip. At start-up, all programmable parameters are loaded onto the chip in less than 1ms; this is the programming overhead when time multiplexing a single detector to detect more object classes. As shown in the system block diagram in Fig. 16, an arbiter controls the DRAM access between the chip, the camera and the display. Pixels from the camera are buffered in the DRAM, and the system can also process fixed pre-loaded frames in the DRAM. The FPGA reads the chip detection output, performs non-maximal suppression to remove overlapping detection boxes, draws the detection boxes on the pixels stream and shows the results on the display through an HDMI interface.

Fig. 17 shows examples of the chip detection results for both modes: live camera feed and fixed frames. DPM filters are trained offline using [25]. In the first row of Fig. 17, the chip is programmed to detect pedestrians in a live video feed. Fig. 17(a) and (c) show the detection robustness with side view and object deformation. In the second row of Fig. 17, the chip is programmed to detect pedestrians and cars in fixed frames. Results show the ability of the chip to detect multi-object classes and multiple instances of the object at different sizes and scales. Fig. 17(f) shows an example of detecting occluded objects (the yellow cab in the center).

The detection accuracy of the proposed architecture is measured on PASCAL VOC 2007 dataset [32], which is a widely used image dataset containing 20 different object classes in 9,963 images. Table II compares our architecture with two CPU implementations[1]: the original DPM V5 [25] and the accelerated version in [27]. Both software implementations use a 6-core Intel Xeon processors with 32GB of memory. Note that the VOC2007 dataset maximum image size is only 500×500. To compare with these other works, our accelerator can made to support the resolution of the VOC dataset images by stitching 4 images into one full HD frame; as a result, the effective frame rate of the proposed accelerator increases by 4× to 240fps for the images in the VOC dataset. The detection accuracy difference changes from one class to the other, but the mean AP numbers compared to the fast DPM implementation in [27] are equal. The error in mean AP of the proposed accelerator can be potentially reduced by re-training the parts and deformation filters with the vector quantized features.

[1]Object detection accuracy is measure by Average Precision (AP), where higher AP means better detection accuracy.

C. Performance Comparison

Table III shows a comparison with a HOG-based object detection accelerator. Both accelerators process full HD videos at 30fps in real-time. The architecture in [11] has two classification engines similar to this work, but it does not have on-chip pyramid generation to support multi-scale and does not detect deformable parts. The two classifiers in [11] can be used to detect two different object classes at a single scale, or support multi-scale detection by processing a 5-level pyramid generated and packed off-chip into two full HD frames. Our proposed architecture supports multi-scale generation and detection on-chip and detects deformable parts, which both increase the detection accuracy and robustness, while consuming 30% less energy/pixel compared to [11].

Table IV shows a comparison between the proposed architecture and our software implementation of the same DPM detection algorithm mapped on ODROID-XU3 board, which is an embedded platform designed for low power consumption. This board is powered by the Samsung Exynos5422 embedded processor and 2GB of memory. The processor uses ARM big.LITTLE technology with Cortex-A15 high performance quad-core and Cortex-A7 low power quad-core. The software implementation uses Halide [33] and OpenMP to achieve a 10× throughput increase. All numbers are measured on full HD frames. Table IV shows that a maximum throughput of 0.24fps can be achieved on ODROID-XU3 board when using the Cortex-A16 four cores. Lower energy can be achieved by using the low power Cortex-A7 four cores which are 2.4× slower, but achieve almost 4× lower energy consumption. The proposed ASIC architecture consumes 3 orders of magnitude less energy than the Exynos embedded processor.

V. Conclusion

In this paper, we present a programmable, energy-efficient and real-time object detection hardware accelerator using deformable parts models for low power and high throughput applications. Detecting deformable parts enables 2× the detection accuracy of traditional rigid body models with 35× increase in the required computation. To overcome this large overhead, we propose classification pruning, vector quantization and feature basis projection to achieve real-time processing at low energy consumption. The chip is implemented in a 65nm CMOS technology, and can process full HD 1920×1080 videos at 30fps while consuming only 58.6mW, resulting in an energy efficiency of 0.94 nJ/pixel and a peak performance of 1168 GOPS/W. The chip has a measured maximum throughput of 60fps at 1.11V with an energy of 1.74 nJ/pixel. The optimizations achieve an overall 5× power reduction and 3.6× smaller memory size. With the reported energy numbers of less than 1 nJ/pixel, this accelerator enables object detection to be as energy-efficient as video compression. This is an important step towards making computer vision as ubiquitous as video compression, which is found in most cameras today.

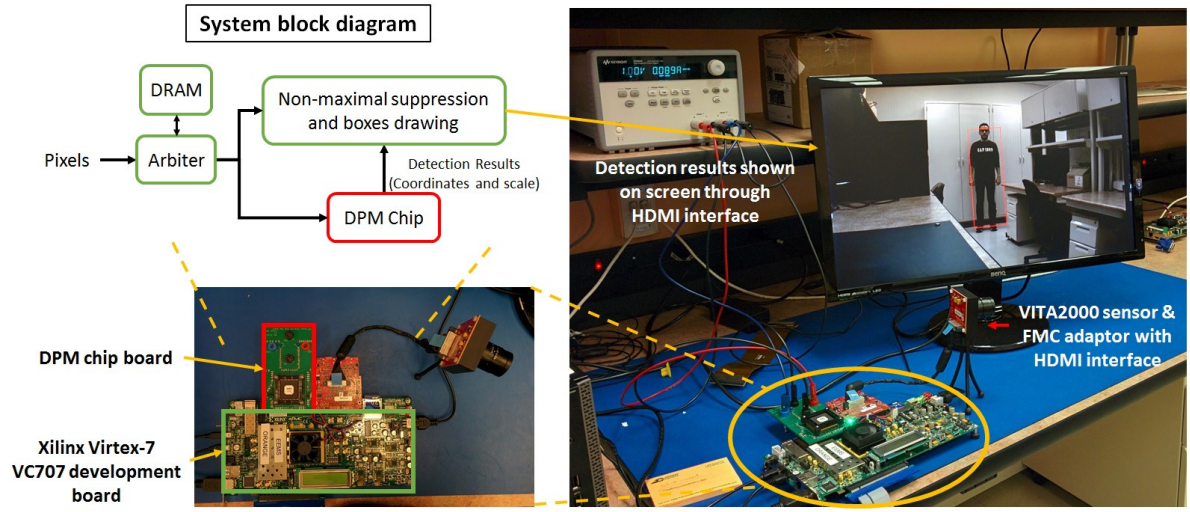


Fig. 16. Demonstration system.

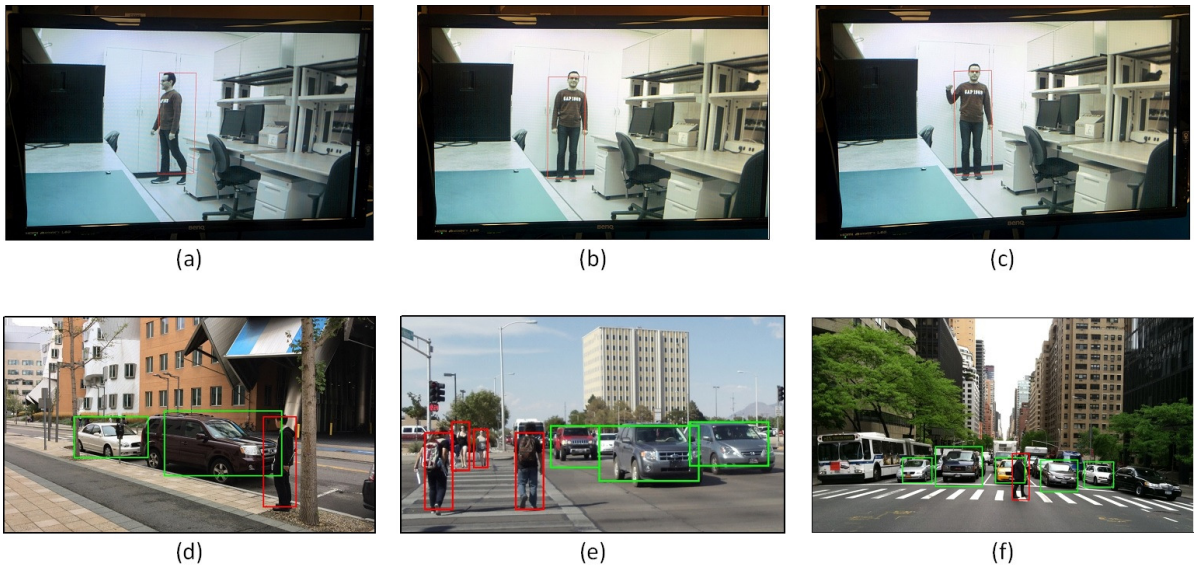


Fig. 17. Chip detection results examples. (a,b,c) Live camera feed input. (d,e,f) Fixed frames pre-loaded to DRAM.

ACKNOWLEDGMENT

The authors would like to thank TSMC University Shuttle Program for the chip fabrication, DARPA and Texas Instruments for funding, and Ariana Eisenstein (MIT M.Eng. 2016) for her help in developing the demonstration system.

REFERENCES

[1] Y. Artan, O. Bulan, R. P. Loce, and P. Paul, "Passenger Compartment Violation Detection in HOV/HOT Lanes," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 395–405, Feb. 2016.

[2] M. Pedersoli, J. Gonzlez, X. Hu, and X. Roca, "Toward Real-Time Pedestrian Detection Based on a Deformable Template Model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 355–364, Feb. 2014.

[3] S. Piao, L. Kiekbusch, D. Schmidt, K. Berns, D. Hering, S. Wirtz, N. Hering, and J. Weiland, "Real-time multi-platform pedestrian detection in a heavy duty driver assistance system," *International Commercial Vehicle Technology Symposium*, March 2016.

[4] T. Moranduzzo, F. Melgani, Y. Bazi, and N. Alajlan, "A fast object detector based on high-order gradients and Gaussian process regression for UAV images," *International Journal of Remote Sensing*, vol. 36, no. 10, pp. 2713–2733, May 2015.

[5] H. Sharma, T. Dutta, V. Adithya, and P. Balamuralidhar, "A Real-Time Framework for Detection of Long Linear Infrastructural Objects in Aerial Imagery," *International Conference on Image Analysis and Recognition (ICIAR)*, pp. 71–81, July 2015.

[6] J. Y. Kim, M. Kim, S. Lee, J. Oh, K. Kim, and H. J. Yoo, "A 201.4 GOPS 496 mW Real-Time Multi-Object Recognition Processor With Bio-Inspired Neural Perception Engine," *IEEE J. Solid-State Circuits*, vol. 45, no. 1, pp. 32–45, Jan. 2010.

TABLE II. DETECTION ACCURACY NUMBERS FOR PASCAL VOC 2007 DATASET [32].

Method	**DPM V5 [25]**	**Fast DPM [27]**	**This work**
aeroplane	0.3318	0.2695	0.2104
bicycle	0.5878	0.5735	0.5242
bird	0.1019	0.0909	0.0998
boat	0.1801	0.0303	0.0942
bottle	0.2535	0.1938	0.2208
bus	0.5056	0.4130	0.3911
car	0.5271	0.4240	0.4583
cat	0.1904	0.1725	0.1372
chair	0.2046	0.0909	0.1298
cow	0.2444	0.1062	0.2044
diningtable	0.2750	0.2500	0.1667
dog	0.1238	0.1159	0.1073
horse	0.5709	0.4735	0.4979
motorbike	0.4838	0.3850	0.4024
person	0.4327	0.3736	0.3391
pottedplant	0.1366	0.1179	0.1047
sheep	0.2154	0.0909	0.1274
sofa	0.3633	0.2860	0.2588
train	0.4651	0.3962	0.3602
tvmonitor	0.3943	0.3703	0.3462
mean AP	**0.33**	**0.26**	**0.26**
Platform	**Intel Xeon 6 cores**	**Intel Xeon 6 cores**	**ASIC in 65nm**
Throughput (fps)	**0.07**	**30**	**240**

TABLE III. CHIP PERFORMANCE COMPARISON

	JSPS [11]	**This work**
Process	65nm	65nm
Chip size	4.2×2.1 mm^2	4.0×4.0 mm^2
Input resolution	1920×1080	1920×1080
Multi-scale	Single scale	12-level pyramid
Deformable parts	No	8 parts
Object classes	2	2
Frame rate	30	30
Frequency	84.3 MHz	62.5 MHz
Power	84mW	58.6mW
Energy/pixel	1.35 nJ	0.94 nJ

[7] I. Hong, K. Bong, D. Shin, S. Park, K. J. Lee, Y. Kim, and H. J. Yoo, "A 2.71 nJ/Pixel Gaze-Activated Object Recognition System for Low-Power Mobile Smart Glasses," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 45–55, Jan. 2016.

[8] Y.-H. Chen, T. Krishna, J. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE J. Solid-State Circuits*, pp. 262–264, Feb. 2016.

[9] A. Suleiman, and V. Sze, "An Energy-Efficient Hardware Implementation of HOG-Based Object Detection at 1080HD 60 fps with Multi-Scale Support," *J. Signal Processing Systems*, vol. 84, no. 3, pp. 325–337, Sep. 2016.

[10] A. Suleiman, Z. Zhang, and V. Sze, "A 58.6mW Real-Time Programmable Object Detector with Multi-Scale Multi-Object Support Using Deformable Parts Model on 1920x1080 Video at 30fps," *in IEEE Symposia on VLSI Technology and Circuits*, June 2016, pp. 184–185.

[11] K. Takagi, K. Tanaka, S. Izumi, H. Kawaguchi, and M. Yoshimoto, "A Real-time Scalable Object Detection System using Low-power HOG Accelerator VLSI," *J. Signal Processing Systems*, vol. 76, no. 3, pp. 261-274, Sep. 2014.

[12] K. J. Lee, K. Bong, C. Kim, J. Jang, H. Kim, J. Lee, K. R. Lee, G. Kim, and H. J. Yoo, "A 502GOPS and 0.984mW dual-mode ADAS SoC with RNN-FIS engine for intention prediction in automotive black-box system," *in IEEE ISSCC Dig. Tech. Papers*, Feb. 2016, pp. 256–257.

[13] J. Tanabe, S. Toru, Y. Yamada, T. Watanabe, M. Okumura, M. Nishiyama, T. Nomura, K. Oma, N. Sato, M. Banno, H. Hayashi, and T. Miyamor, "A 1.9TOPS and 564GOPS/W heterogeneous multicore SoC with color-based object classification accelerator for image-recognition applications," *in IEEE ISSCC Dig. Tech. Papers*, Feb. 2015, pp. 1–3.

[14] J. Oh, G. Kim, J. Park, I. Hong, S. Lee, and H. J. Yoo, "A 320mW 342GOPS real-time moving object recognition processor for HD 720p video streams," *in IEEE ISSCC Dig. Tech. Papers*, Feb. 2012, pp. 220–222.

[15] M. Meingast, C. Geyer, and S. Sustry, "Vision based terrain recovery for landing unmanned aerial vehicles," *in IEEE Conference on Decision and Control*, vol. 2, pp. 1670–1675, Dec. 2004.

[16] B. A. Myers, J. H. Burns, and J. M. Ratell, "Embedded Electronics in Electro-Mechanical Systems for Automotive Applications," *SAE Technical Paper*, 2001.

[17] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *IEEE Transaction on PAMI*, vol. 23, no. 4, pp. 349-361, Apr. 2001.

[18] P. Viola, and M. Jones, "Rapid object detection using a boosted cascade of simple features," *in IEEE CVPR*, June 2001, vol. 1, pp. 511-518.

[19] N. Dalal, and B. Triggs, "Histograms of oriented gradients for human detection," *in IEEE CVPR*, June 2005, vol. 1, pp. 886-893.

[20] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models," *IEEE Transaction on PAMI*, vol. 32, no. 9, pp. 1627–1645, Sept. 2010.

TABLE IV. COMPARISON WITH ODROID-XU3 BOARD PROCESSING FULL HD 1920×1080 FRAMES

Platform	Cortex-A7		Cortex-A15		This work	
	1 core	4 cores	1 core	4 cores	0.77V	1.11V
Process Technology	28nm HKMG		28nm HKMG		65nm CMOS	
Throughput (fps)	0.04	0.10	0.11	0.24	30	60
Power (mW)	155.6	383.5	1,703.8	3,575.6	58.6	216.5
Energy (nJ/pixel)	1,881.8	1,849.0	7,301.2	7,165.7	0.94	1.74

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *in IEEE NIPS*, Dec. 2012.

[22] P. Dollar, S. Belongie, and P. Perona, "The Fastest Pedestrian Detector in the West," *in Conference of the British Machine Vision*, Sept. 2010, pp. 68.1-68.11.

[23] C. C. Ju *et al.*, "A 0.5 nJ/Pixel 4 K H.265/HEVC Codec LSI for Multi-Format Smartphone Applications," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 56–67, Jan. 2016.

[24] R. Benenson, M. Omran, J. Hosang, and B. Schiele, "Ten Years of Pedestrian Detection, What Have We Learned?," *in IEEE ECCV*, Sept. 2014, pp. 613–627.

[25] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester, "Discriminatively Trained Deformable Part Models, Release 5," In http://people.cs.uchicago.edu/ rbg/latent-release5/.

[26] C. Cortes, and V. Vapnik, "Support-Vector Networks," *J. of Machine Learning*, vol. 20, no. 3, pp. 273-297, Sept. 1995.

[27] M. A. Sadeghi, and D. Forsyth, "30Hz Object Detection with DPM V5," *in IEEE ECCV*, Sept. 2014, pp. 65–79.

[28] M. Hirabayashi, S. Kato, M. Edahiro, K. Takeda, and S. Mita, "Accelerated Deformable Part Models on GPUs," *in IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 6, pp. 1589–1602, June 2016.

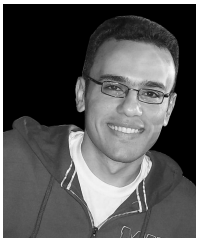
[29] "VITA 2000 2.3 Megapixel 92 FPS Global Shutter CMOS Image Sensor," "http://www.onsemi.com/PowerSolutions/product.do?id=VITA2000"

[30] P. Felzenszwalb *et al.*, "Cascade object detection with deformable part models," *in IEEE CVPR*, June 2010.

[31] M. Sadeghi, and D. Forsyth, "Fast template evaluation with Vector Quantization," *in IEEE NIPS*, Dec. 2013.

[32] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results," "http://host.robots.ox.ac.uk:8080/pascal/VOC/voc2007/"

[33] J. Ragan-Kelley, C. Barnes, A. Adams, S. Paris, F. Durand, and S. Amarasinghe, "Halide: A Language and Compiler for Optimizing Parallelism, Locality, and Recomputation in Image Processing Pipelines," *ACM J. SIGPLAN Not.*, vol. 48, no. 6, pp. 519–530, June 2013.



Amr Suleiman (S'08) received the B.S. and M.S. degrees in Electronics and Electrical Communications Engineering from Cairo University, Egypt in 2008 and 2011 respectively. He received the M.S. degree in Electrical Engineering from MIT in 2013. He is currently working toward his Ph.D. under the supervision of Prof. Vivienne Sze. Amr's work focuses on developing new energy-efficient implementations for machine vision algorithms (e.g. detection, recognition, and tracking). Amr is a recipient of the Endowed fellowship of the Arab Republic of Egypt, and the 2015 Broadcom Foundation University Research Competition.

Zhengdong Zhang (S'15) received the B.S. in Computer Science in 2011 from Tsinghua University, Beijing, China. He received the M.S. degree in Computer Science from Massachusetts Institute of Technology, Cambridge in 2014. Between 2011 and 2012 he worked in Microsoft Research Asia, Beijing, China, as an Assistant Researcher. He is pursuing the Ph.D. degree under the supervision of Prof. Vivienne Sze. His research interest spans the area of sparsity, low-rank matrix recovery, symmetry/regularity of textures, 3D computer vision, computational photography and vision systems. His current research focuses on the design of energy-efficient vision systems.

Vivienne Sze (S'04–M'10–SM'16) received the B.A.Sc. (Hons) degree from the University of Toronto, Toronto, ON, Canada, in 2004, and the S.M. and Ph.D. degree from the Massachusetts Institute of Technology (MIT), Cambridge, MA, in 2006 and 2010, respectively, all in electrical engineering.

She has been an Assistant Professor with MIT in the Electrical Engineering and Computer Science Department since August 2013. Her research interests include energy-aware signal processing algorithms and low-power circuit and system design for portable multimedia applications. Prior to joining MIT, she was a Member of Technical Staff with the Systems and Applications R&D Center, Texas Instruments (TI), Dallas, TX, USA, where she designed low-power algorithms and architectures for video coding. She also represented TI at the international JCT-VC standardization body developing HEVC. Within the committee, she was the primary coordinator of the core experiment on coefficient scanning and coding, and has chaired/vice-chaired several ad hoc groups on entropy coding. She is a coeditor of *High Efficiency Video Coding (HEVC): Algorithms and Architectures* (Springer, 2014).

Prof. Sze was a recipient of the 2016 AFOSR Young Investigator Research Program (YIP) Award, 2016 3M Non-Tenured Faculty Award, 2014 DARPA Young Faculty Award, 2007 DAC/ISSCC Student Design Contest Award and a co-recipient of the 2008 A-SSCC Outstanding Design Award. In 2011, she received the Jin-Au Kong Outstanding Doctoral Thesis Prize in Electrical Engineering at MIT. She received the Natural Sciences and Engineering Research Council of Canada (NSERC) Julie Payette fellowship in 2004, the NSERC Postgraduate Scholarships in 2005 and 2007, and the Texas Instruments Graduate Womens Fellowship for Leadership in Microelectronics in 2008.