Depth-Directed Hardware Object Detection

Christos Kyrkou, Christos Ttofis and Theocharis Theocharides,
KIOS Research Center, Department of Electrical and Computer Engineering
University of Cyprus
Nicosia, Cyprus
{kyrkou.christos, ttofis.christos, ttheocharides}@ucy.ac.cy

Abstract— **Object detection is a vital task in several emerging applications, requiring real-time detection frame-rate and low energy consumption for use in embedded and mobile devices. This paper proposes a hardware-based, depth-directed search method for reducing the search space involved in object detection, resulting in significant speed-ups and energy savings. The proposed architecture utilizes the disparity values computed from a stereoscopic camera setup, in an attempt to direct the detection classifier to regions that contain objects of interest. By eliminating large amounts of search data, the proposed system achieves both performance gains and reduced energy consumption. FPGA simulation results indicate performance speedups up to 4.7 times and high energy savings ranging from 41-48%, when compared to the traditional sliding window approach.**

Keywords- Hardware Object Detection; Stereoscopic Disparity Computation; FPGA Image Processing

I. INTRODUCTION

Object detection is an important task in several computer vision and artificial intelligence applications, that refers to the ability of a computer system to determine the presence of objects of interest in images. Emerging applications that utilize object detection such as human-computer interaction, surveillance, biomedical imaging, space missions, and automobile applications among others, require real-time detection and low energy consumption. In such cases, software implementations of object detection usually struggle to meet the real-time performance and low energy constraints, and as a result some form of hardware acceleration or even a complete custom hardware implementation is preferred [16,18]. Additionally, the improved performance stemming from hardware architectures extends the application spectrum in the 3D world, where stereo and multi-view cameras can be used for object detection in 3D environments. Furthermore, the performance in terms of detection frame rates of hardware architectures enables optimization steps to be included in the algorithm, while maintaining the real-time constraints. Such steps include image enhancement techniques, segmentation and other well-known steps traditionally used in software implementations.

Several hardware-based object detection architectures yielding real time results emerged in recent works [17-19]. The majority of these architectures employ the traditional sliding search window approach, which generates a lot of data and makes the object detection process time consuming and tedious in terms of hardware data flow. Moreover, only a few implementations feature optimizations, which have been widely used in software implementations, that can potentially contribute in reducing the hardware constraints in terms of efficient data handling and energy consumption. Some of the few hardware optimizations employed, utilize techniques such as color segmentation and background segmentation [10,12]. Recently, the idea to utilize depth information emerging from disparity maps in stereoscopic applications has been proposed for reducing the effective search space in the input image. Such software implementations have yield promising results [5,9,11].

This paper, motivated by the software mechanism presented in [5], proposes a hardware integration of depth information extracted from stereoscopic disparity map computation, as a means to reduce the search space and energy consumption of hardware object detection systems. This reduces the overall search area of the classifier, which in turn allows real-time detection of larger images as well as reduction of the overall energy consumption, when compared to naive sliding window search used in traditional object detection. In contrast to [5], this paper exploits the inherit parallelism of the disparity map computation and object detection algorithms by integrating an improved hardware disparity computation unit developed from the original model presented in [7] along with a hardware SVM-based object detection classifier developed from the base model presented in [6], to propose an optimized object detection framework for hardware stereoscopic systems. The paper discusses the hardware design parameters and the necessary trade-offs involved in using the disparity information, and presents a complete experimental platform that includes a stereoscopic camera capturing platform, camera calibration, classifier training and visual display. The proposed system was implemented on a Virtex-5 FPGA platform targeting the application of face detection for a stereo image pair of 240x320 (row x column) pixels. We observe frame-rate speedups in the range of 1.9-4.7 when compared to a traditional sliding window approach and estimated energy savings of 41-48%.

The paper is organized as follows. Section II gives a summary of the object detection and stereoscopic disparity computation, and Section III provides information about the related work. Section IV presents the hardware architecture, and Section V presents the experimental platform and simulation results. The paper concludes by giving future work directives in Section VI.

This work was supported in part by the Cyprus Research Promotion Foundation under contract ΤΠΕ/ΠΛΗΡΟ/0308(ΒΙΕ)/04.

978-3-9810801-7-9/DATE11/©2011 EDAA

II. BACKROUND

A. Object Detection Overview

The process of image object detection deals with determining whether an object of interest is present in an image/video frame or not, regardless of its size, orientation, and environmental conditions. An image object detection system receives an input image/video frame, and subsequently searches to locate possible objects of interest. This search is usually done by extracting smaller regions from the frame, called search windows, of *m x n* pixels, which are processed by a classification algorithm to determine if they belong to the object of interest or not. To account for the variable sizes in objects given the typically fixed size of the search window, an object detection system usually downscales the input image in steps, effectively reducing the size of the object of interest, and reexamines the image, until the downscaled image is equal to the size of the search window. Many downscaled images are produced from a single input image/video frame, each in turn producing a number of search windows, which increases the amount of data that must be processed by the classification algorithm. While exhaustive search has been the most popular approach, the number of search windows becomes prohibitive as the size of the input image increases. Popular methods that have been used to reduce the number of generated windows include skin segmentation, background removal and recently depth information.

B. Depth Extraction

Depth information can be extracted by using a stereo vision system which works similarly to the way that the human visual system infers depth. Stereo vision systems can infer depth information about a scene from a stereo image pair (usually referred to as left and right images) [14-15]. Depth information evolves from the computation of the disparity map, from which information about the depth of objects in an image frame, relative to the location of the camera(s), can be extracted. Generally, there are three important tasks for computing the disparity map: *camera calibration*, *stereo image rectification* and *stereo correspondence*. Calibration integrates information from intrinsic camera parameters such as focal length (*f*), and extrinsic parameters such as relative orientation angles between the two cameras, to determine the camera perspective projection matrices, necessary for the stereo rectification algorithm. Rectification can project one of the two images in the other's common plane to reduce a 2D search problem in non-rectified images into a 1D search problem along the horizontal raster lines of the rectified images [14]. The stereo correspondence algorithm takes the two rectified images as input and produces a disparity map *d(x,y)* for each pixel in one of the two images. Generally, the disparity map stores distances between corresponding points of interest in the two images, and is computed using searching and matching techniques. Depth information (Z) can be computed from the disparity map using the formula $Z = f * (b/d)$ where b refers to the baseline distance between the stereo camera optical centers.

C. Depth Guided Object Detection

By using depth information extracted from a pair of stereo images it is possible to estimate the size of the object at a given location, thus avoiding the downscaling process and subsequently reducing the generation of a large number of search windows from each downscaled version. The relationship between the actual size of the object (O_{size}) as is represented in the real 3D world and its projection in one of the stereo image frames (W_{size}) can be estimated using equation (1) [14]. The equation essentially relates the size of the object in the camera image plane with its size in the actual image. Using the relationship between depth and the disparity map (d) we can use the disparity values instead (2), thus, avoiding the direct computation of depth.

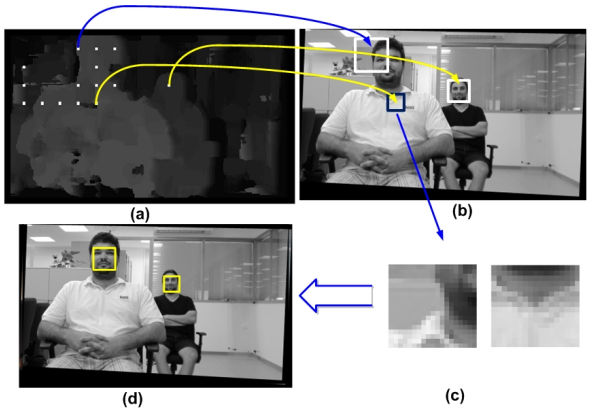

Figure 1. Window Size Estimation Algorithm: (a) The disparity map is sampled every few pixels (b) For each disparity value the corresponding window size is estimated (c) Read window pixel values from one of the two stereo images (d) Output result of classified windows.

$$(\mathrm{W}_{\mathrm{size}}/f) = (O_{size}/Z) \Rightarrow \mathrm{W}_{\mathrm{size}} = f * (O_{size}/Z) \quad (1)$$

$$Z = f * (b/d) \Rightarrow \mathrm{W}_{\mathrm{size}} = (O_{size}/b) * d \quad (2)$$

The depth-directed search overall procedure is given in Fig.1. Each value in the disparity map corresponds to a certain search window size and the coordinates of that value are at the center of that window. Consequently, there is no need for exhaustive search in the disparity map on a pixel by pixel basis, but every few pixels, the same way that the sliding window operates in traditional object detection schemes. Furthermore, if a large object is found at the border of the image the window size it will most likely exceed the image dimensions. That window can then be discarded and that object is left to be detected in another window. Finally, any disparity values that map to windows which are smaller than the expected window size are discarded and the next disparity value is processed. In this way the number of candidate search windows is further reduced compared to the traditional object detection search method.

III. RELATED WORK

In recent years, a fair amount of work has been done in hardware implementations of object detection algorithms. These works utilize techniques such as neural networks [16-17] and the Viola Jones detection algorithm [18-19], and achieve relatively high detection frame rates. However, the majority of these implementations follow the sliding window search approach, where as mentioned earlier, a lot of windows investigated do not contain object information, and as such a

large amount of energy is wasted. Moreover, as the input image size increases, this problem is further emphasized.

The use of optimization techniques, such as skin color segmentation and background removal as a means to reduce the number of windows to be processed, and increase the resulting frame rate, is suggested in several software implementations [10,20,21], with only a few hardware implementations reported thus far [16,22]. The former technique, however, can only be applied to applications related to skin-based detection such as face detection, while the latter may have increased complexity and computational demands depending on the detection scenario. Alternatively, depth information as a means to reduce the search space has recently been proposed in [5]. Depth is interpreted in the same way in many environments and thus provides a more efficient way to reduce the search space for different applications. Depth information can also be used after the detection process to reduce the false positive detections, as shown in [9]. The work done in [9], however, targets improved classification accuracy rather than reducing the search space to increase the performance.

To the best of our knowledge, this is the first paper that investigates the use of depth as search reduction mechanism in hardware for performance improvement and energy reduction. There has been some initial work done in software [5,10,11]. The work done in [5] demonstrated the performance speedups from reducing the search space. The authors developed a methodology that uses depth information to estimate the size of each search window, which is then fed to a Viola-Jones face detector. However, the approach presented in that work is only suitable for video sequences where the scene does not change much between successive frames, otherwise the frame rate drops significantly. Moreover, the Viola-Jones classifier requires many hardware resources [18] so a more flexible and simple classifier might be appropriate. In this work, we attempt to optimize the depth based algorithm and implement it in hardware, to allow for scalability and performance optimizations, and potentially energy reduction. The proposed hardware implementation attempts to search a larger part of the depth information compared to [5] by utilizing parallel hardware accesses to the depth information memory. We integrate the depth estimation approach into an existing object detection hardware implementation, in order to explore the potential performance and energy consumption benefits.

IV. Proposed Hardware Architecture

The proposed architecture consists of three major hardware units; the Disparity Extraction Unit (DEU), the Window Extraction Unit (WEU) and the Classification Engine (CE). The system also consists of a system controller that optimizes accesses to the external memory, controls I/O operation, and synchronizes the other major units. The system uses two on-chip buffers to temporarily store the image data; the first buffer stores the disparity values used for the search, and the second buffer stores the parts of the image that are being searched for potential objects by the CE. Fig. 2 shows an overview of the system architecture and the communication flow between units.

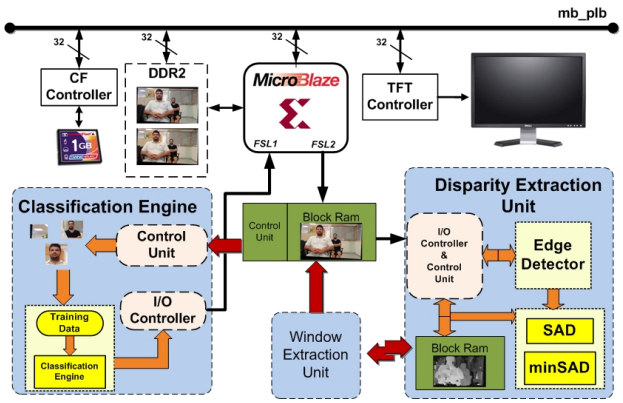

Figure 2. Proposed Hardware System Architecture

A. Disparity Extraction Unit

The Disparity Extraction Unit (DEU) integrated to the system is based on an improved version of the hardware architecture that was proposed in [7]. This DEU combines the SAD matching algorithm with the features (edges) extracted by the use of an edge detector in order to perform correlation on rectified stereo image pairs that present a maximum disparity range of 40 pixels (maximum correlation window sizes up to 11x11). The improved version of the architecture also uses optimized memory accesses to the external memory and was designed with emphasis on parallelism. These features enable the architecture to obtain frame rates that exceed the 150 fps for an image size of 240x320. This is particularly important, as the time for disparity computation must be small enough in order to obtain a speedup in the overall operation (disparity computation + object detection).

B. Window Extraction Unit

The window extraction unit (WEU), shown in Fig. 3, is responsible for computing (2) which estimates the size of the disparity-directed search window, based on the disparity values provided by the DEU. It essentially links the DEU with the classification engine by generating the addresses for the pixels that will be fed to the classification engine. The WEU receives disparity values and the corresponding coordinates of those values from the DEU. It then estimates the necessary search window size in the original input image. The classification stage usually receives fixed-size search windows (*m x m* in our case), therefore the WEU is responsible for compensating for the different sizes. The WEU performs comparisons to check if the disparity-directed search window is smaller than the classifier window size (*m x m*), and whether it exceeds the original input image boundaries. If one of the conditions is true, the disparity-directed search window is discarded.

In the likely case that the window size is greater than *m x m* but smaller than the original image boundaries, we need to downscale the window to *m x m* so that it fits in the classification stage. We employ the downscaling method as it does not increase the memory and computation demands, even if it can lead to some data loss. Furthermore, in this case we do not need to read the entire disparity-directed search window first and then downscaling it, but rather read only the pixels that are required to form the *m x m* search window from the larger window. This ensures that for every window, regardless

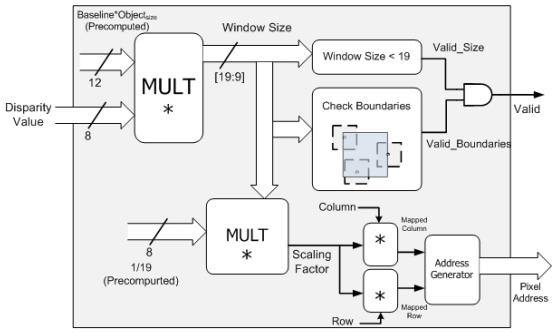

Figure 3. Window Extraction Unit

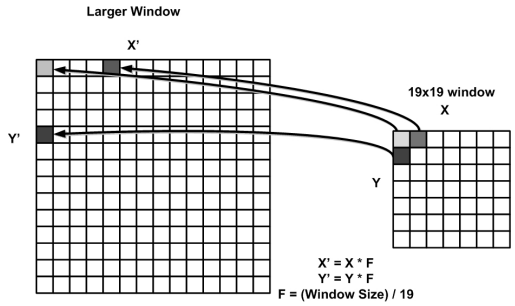

Figure 4. Downscaling Process: Each coordinate in the 19x19 window is mapped to a coordinate in the larger window. The coordinates where the 19x19 window coordinates are mapped, correspond to the pixels values that will be read for classification.

of the estimated window size, only m^2 pixel values will be read and sent to the CE. To achieve this, we map every coordinate from a *m x m* window onto one coordinate of the larger window, by essentially upscaling the *m x m* window and fetching the pixels from the mapped coordinates. Details of this scaling technique are given in Fig. 4.

C. Classification Engine

The classification engine (CE) used in the proposed detection system features a simple, yet powerful Support Vector Machine [3] hardware architecture proposed in [6]. The processing elements are responsible for computing dot-product operations between vectors while dedicated units handle the scalar processing of the SVM computation flow. The classifier is modular and simple; hence, a number of parallel classification units can be used, to account for different performance demands. The classifier receives the *m x m* search window as input, and returns whether the window contains the object of interest or not. The CE architecture and the interconnectivity between processing elements are illustrated in Fig. 5.

D. Flow of Data

The overall operation is partitioned into three stages: disparity computation, search window estimation and window classification. The memory controller fetches the stereo image pair from the external memory to the DEU in raster-scan fashion, and stores the incoming pixels in the *input image buffer*. The DEU utilizes these values from the buffer and computes the disparity map, which in turn stores in the *disparity buffer*. Then the WEU can begin the window estimation procedure for the computed disparity values. If the generated window is valid the WEU generates the addresses

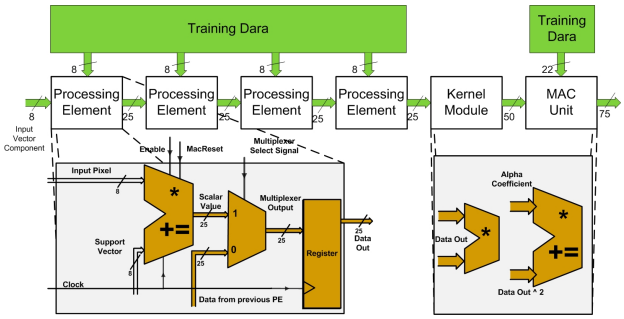

Figure 5. SVM classificattion engine acrhicture

from which to fetch pixels for that window. The memory controller receives the addresses and starts fetching the pixel data from the *input image buffer*. The incoming pixels are fed to the CE which determines whether the object of interest is present in the window or not.

V. Experimental Platform and Results

A. Experimental Platform and Methodology

To evaluate the proposed architecture we implemented the system architecture shown in Fig. 2 using a Xilinx ML505 board (Virtex 5-LX110T FPGA), targeting face detection, a popular object detection application. The Microblaze soft processor [8] was used as the system I/O controller to handle tasks such as memory transfers and external I/O. We used a custom built stereo vision system to construct a dataset consisting of ten 240x320 stereo image pairs to evaluate the architecture. The stereo system consists of two video cameras separated by a baseline distance of 77mm, both with a focal length of 25mm. The stereo system was calibrated using the Camera Calibration Toolbox for MATLAB [13], and was then used to capture stereo image pairs, which were rectified and stored on the on-board DRAM, and used as input data to the system. Visual output was directed to a digital monitor, through the on-board DVI output. Fig. 6 (b and c) shows the output of our experimental framework.

The training of the Support Vector Machine classifier was carried out in MATLAB using the methodology and strategies employed in [1,2]. We used the face detection database from [4], which we also enhanced using bootstrapping. The training procedure produced 400 support vectors for a 2nd degree polynomial kernel $(x \cdot y + 1)^2$ [3], and all training data were stored on the FPGA Block RAM.

To evaluate the performance of the proposed architecture we setup two different configurations: one using the disparity map to guide the search window space, and one using the traditional sliding window approach. The latter consists only of the SVM classifier and a controller which fetches window data from the input image for various scales. The two systems have the same I/O constraints, memory requirements, and training data. We identify the speedup and hardware overheads of the proposed architecture compared to the traditional object detection approach. Additionally, we also provide results when increasing the number of CEs from one to three, to illustrate the impact of inter-window parallelism as an added benefit of the reduction in search data stemming from the disparity-directed computation.

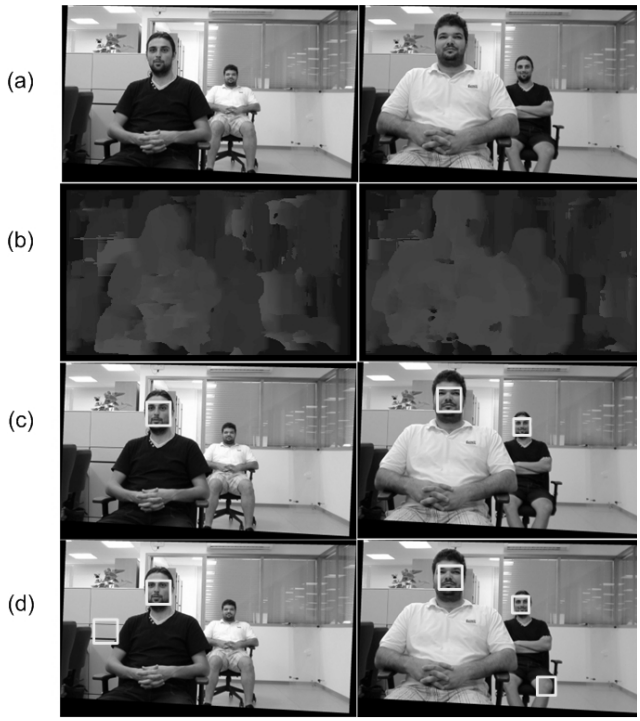

Figure 6. Evaluation Images and Results: (a) Right Images from stereoscopic pairs (b) Disparity Maps from stereo processing (c) Detection results using depth-guided method (d) Detection results using the traditional sliding window approach

The sliding window approach requires that the input 240x320 is scaled a number of times to account for different object sizes. As such the input 240x320 image is downscaled five times resulting in additional five images (180x240, 90x120, 45x60, 36x48, 19x19). For each image we generate windows with an overlap of 5 pixels resulting in a total of 4,874 generated windows. On the other hand, for the depth-directed approach we sample the disparity map every 5 pixels to achieve a high granularity and also compensate for erroneous disparity values. In the worst case scenario all sampled disparity map windows will be valid windows resulting in a total of 2496 windows. However, practically this will not be the case, as from a number of experiments we carried out the number of valid windows is on average around 1000, and possibly less depending on the number and size of objects inside the input image.

The performance of depth accelerated object detection system is measured by the time necessary to compute the disparity map of the input stereo pair and the time needed by the classification engine (in our case SVM) to classify the generated windows from one of the two stereo images. The total processing time is affected by the performance of the DEU and the CE. When the DEU performance is greater, the speedup will also be improved. On the other hand, when the performance of the CE is greater, the speedup is constraint by the DEU performance. The experimental results over the input stereo pairs of images we used indicate that by using the depth guided object detection the number of generated windows is reduced on an average of 80% (about 3800 less windows) compared to the traditional sliding window approach. As a result, the frame rate is also increased.

TABLE I. PERFORMANCE FOR DIFFERENT SYSTEM CONFIGURATIONS

Configuration			FPS	# of Windows
# of CE	*Window Generation Method*	*# of WEU*		
1	SW	-	11	4874
2	SW	-	21	4874
3	SW	-	32	4874
1	DA	1	21	2496 (worst case)
			53	~1000(average)
2	DA	2	42	2496 (worst case)
			107	~1000 (average)
3	DA	3	64	2496 (worst case)
			150	~1000 (average)

SW: Sliding Window, DA: Depth Accelerated, CE: Classification Engine

This drastic reduction in search windows makes the performance of the chosen classification stage (CE unit) critical in the proposed implementation. However, this is an easily addressable issue, as the classification engine is modular and scalable, thus additional classification engines can be integrated to the system for performance improvement purposes.

B. Performance Results

We compare the performance speedups when using the depth-directed search compared to the traditional sliding window search, for a range of 1 to 3 CEs. In all three cases, we observe speedups in the range of 1.9 for the worst case scenario, and 4.7 for the average case. The comparisons are done using the same number of classifiers, with and without the integration of depth acceleration. Three classification engines are needed for real time performance. However, it was observed that when a disparity-directed search is used, only one classifier is necessary for real-time performance (for the average case), yielding both hardware and energy savings. It must be noted that in the current configuration, the bottleneck of the system is the classifier. However, as the number of classifiers increases, the performance bottleneck shifts from the classifier to the disparity computation.. Table I summarizes the performance results for various system configurations for both the average and worst case.

In comparison, the algorithm proposed in [5] and implemented in software offers speedup of 2.8, as it does not take advantage of the inherent parallelism of the application. Furthermore, the resulting speedup is also achieved by searching specific regions in the image every two frames instead of one. It must also be noted that [5] uses a different classifier, that consumes many resources when implemented in hardware [18]. Moreover, the detection results are comparable with those of existing hardware detection systems featuring other classifiers [17-19], achieving 28-30 frames per second. We observe that in the worst case with one classifier and depth acceleration the performance is comparable with exiting works, while all other scenarios that use depth acceleration outperform the performance of existing works.

Detection accuracies of the SVM CE are similar to other reported works [1-2] at about 95% for the given training set [4]. While the accuracy of the detector relies heavily on the chosen classifier and the training set and is beyond the scope of this paper, an inherent advantage of the proposed depth-

TABLE II. HARDWARE REQUIREMENTS FOR EACH UNIT AND DIFFERENT SYSTEM CONFIGURATIONS

System	FPGA Resources			
	Slice LUTs (69,120)	*Slice Registers (69,120)*	*DSP48E (64)*	*Block Ram (148)*
WEU	419 (~1%)	33 (~1%)	-	-
CE	14,385 (20%)	5,197 (7.5%)	8 (%)	100 (67%)
DEU	11,996 (17%)	16,145 (23%)	-	-
Microblaze	7,016 (10%)	8,180 (11%)	3 (4%)	30 (20%)
1 CE, SW	21,401 (32%)	13,377 (19%)	8 (12%)	130 (87%)
2 CE, SW	35,786 (51%)	18,574 (26%)	16 (25%)	130 (87%)
3 CE, SW	50,171 (72%)	23,771 (34%)	24 (37%)	130 (87%)
1 CE, DA	33,816 (48%)	29,555 (42%)	8 (12%)	130 (87%)
2 CE, DA	48,201 (69%)	34,752 (50%)	16 (25%)	130 (87%)
3 CE, DA	62,586 (95%)	39,949 (57%)	24 (37%)	130 (87%)

SW: Sliding Window, DA: Depth Accelerated, CE: Classification Engine

directed approach also includes a reduction in the number of false-positive detections, as expected. Evidenced by Fig. 6 (c) and Fig. 6 (d), when utilizing depth information, the number of samples is reduced and as a result the number of false positive detections is reduced implicitly, improving the system accuracy.

C. *Energy Savings and Hardware Overheads*

We used the Xilinx X-Power Analyzer tool and the input stereo data as input to the system in order to determine the relative energy savings when comparing the sliding window approach vs. the depth-directed approach. We obtained average dynamic power consumption figures, which we then used to estimate the total energy consumption per frame for each system. Results show that there is approximately 48% energy reduction in energy per frame when using 1 CE. Energy savings are observed also as the number of CEs increases; when increasing the number of CEs from 1 to 3, the reduction in energy is 41%. We observe that the savings are reduced however, indicating that the primary source of energy consumption in the optimized system is the CE unit. Overall, the energy savings are attributed to the large amount of data that is eliminated from the classification process, which compensates for the disparity computation overheads.

The relative hardware requirements for each major unit are shown in Table II, along with the total hardware requirements per system configuration (number of classification engines and search method). While the area overhead is not negligible the performance speedups are more than enough to justify these overheads.

VI. CONCLUSION

This paper presented an optimized object detection system for stereoscopic applications, where depth information is used to reduce the search space and improve the performance of the system, while yielding significant energy savings. Results show that the performance speedup ranges from 1.9 to 4.7, with energy savings of 41% - 48%. We plan on integrating the depth-directed approach using alternative classification engines and further explore the impact of algorithm specific parameters. Furthermore, the FPGA prototype will be extended to a full-custom ASIC design in order to evaluate performance and energy consumption. Finally, we plan on comparing the depth directed approach with alternative ways of reducing the search spaces such as edge detection.

REFERENCES

[1] E. Osuna, R. Freund, and F. Girosi, “Training support vector machines: an application to face detection,” IEEE Conference on Computer Vision and Pattern Recognition, 1997, pp. 130-136.

[2] B. Heisele, T. Poggio, M. Pontil, " Face Detection in Still GrayImages," unpublished.

[3] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, 1998, pp. 121-16.

[4] MIT Center for Biological and Computation Learning, “CBCL Face Database #1”, Jan 2010. [Online]. Available: http://cbcl.mit.edu/software-datasets/FaceData2.html

[5] H. Wu, K. Suzuki, T. Wada, Q. Chen, "Accelerating Face Detection by Using Depth Information," 3rd Pacific Rim Symposium on Advances in Image and Video Technology, Japan, January 2009, pp. 13-16.

[6] C. Kyrkou, and T. Theocharides, “SCoPE: Towards a Systolic Array for SVM Object Detection, ” IEEE Embedded System Letters, vol. 1, no. 2, pp. 46-49, August 2009.

[7] S. Hadjitheophanous, C. Ttofis, A.S. Georghiades, T. Theocharides, “Towards hardware stereoscopic 3D reconstruction a real-time FPGA computation of the disparity map,” Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010 , vol., no., pp.1743-1748, 8-12 March 2010.

[8] Xilinx, San Jose, CA, “Microblaze Soft Processor”, [Online]. Available: http://www.xilinx.com/tools/microblaze.htm

[9] S. Kosov, K. Scherbaum, K. Faber, T. Thormahlen, H-P Seidel, "Rapid stereo-vision enhanced face detection," 16th IEEE International Conference on Image Processing, 2009 , pp.1221-1224.

[10] T. Darrell, G. Gordon, M. Harville, J. Woodfill, "Integrated Person Tracking Using Stereo, Color, and Pattern Detection," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1998, pp.601.

[11] Y.G. Wang, E.T. Lim, R. Venkateswarlu, "Stereo head/face detection and tracking," International Conference on Image Processing, vol.1, 2004, pp. 605- 608.

[12] Shireen, Khaled, and Sumaya, "Moving object detection in spatial domain using background removal techniques - state-of-art," Recent Patents on Computer Science, vol. 1, 2008, pp. 32-34.

[13] Camera Calibration Toolbox for Matlab [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/

[14] E.Trucco, A. Verri, *Introductory Techniques For 3-D Computer Vision*, Upper Saddle River, NJ, USA, Prentice Hall PTR, 1998.

[15] B. Cyganek, J. Paul Siebert, *Introduction to 3D Computer Vision Techniques and Algorithms*, Wiley, John & Sons, Incorporated, 2009.

[16] M. S. Sadri et al, “An FPGA Based Fast Face Detector,” *In Global Signal Processing Expo and Conf.*, 2004.

[17] Rob McCready, "Real-Time Face Detection on a Configurable Hardware System," *10th International Workshop on Field-Programmable Logic and Applications*, 2000, pp.157-162.

[18] J. Cho, S. Mirzaei, J. Oberg, and R. Kastner, "Fpga-based face detection system using haar classifiers," *Proceeding of the ACM/SIGDA international symposium on Field programmable gate arrays*, New York, NY, USA: ACM, 2009, pp. 103-112.

[19] M. Hiromoto, H. Sugano, and R. Miyamoto, "Partially Parallel Architecture for Adaboost-Based Detection with Haar-Like Features," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol.19, No.1, Jan. 2009, pp.41-52.

[20] Peer, P., Kovac, J., And Solina, Human Skin Color Clustering for Face Detection , EUROCON International Conference on Computer as Tool, 2003,pp. 144-148.

[21] Hwei-Jen Lin, Shwu-Huey Yen, Jih-Pin Yeh, Meng-Ju Lin, "Face Detection Based on Skin Color Segmentation and SVM Classification," ssiri, pp.230-231, 2008 Second International Conference on Secure System Integration and Reliability Improvement, 2008.

[22] Chun He; Papakonstantinou, A.; Deming Chen; , "A novel SoC architecture on FPGA for ultra fast face detection," Computer Design, 2009. ICCD 2009. IEEE International Conference on , vol., no., pp.412-418, 4-7 Oct. 2009