FPGA Implementation of the Histogram of Oriented 4D Surface for Real-Time Human Activity Recognition

Amin Safaei, Q. M. Jonathan Wu (Senior Member, IEEE)
Department of Electrical and Computer Engineering
University of Windsor
Windsor, Ontario N9B 3P4 Canada
Email: {safaeia,jwu}@uwindsor.ca

***Abstract*—This paper presents a system-on-chip field gate programmable array (FPGA)-based, real-time video processing platform for human activity recognition in 3D scenes. The study details the hardware implementation of real-time human action recognition in 3D scenes, with the idea of iterative computing of a histogram of oriented 4D surface normal with an FPGA circuit. Recently, the rapid growth of modern application-based computer vision algorithms has further improved research and implementation. Especially, various accelerators such as histogram-oriented gradient (HOG), histogram-oriented flow (HOF), and histogram-oriented gradient depth (HOD) have been proposed based on the FPGA platform because it has the advantages of high performance and parallel operation. Although proposed FPGA accelerators have good performance in 2D scenes, the accelerator design space has not been well-exploited in 3D scenes. This study details the hardware implementation of a real-time human action recognition algorithm in 3D scenes, including the capture, processing, and display stages.**

***Keywords*—Real time video processing, field programmable gate array, embedded vision, action recognition.**

I. Introduction

Recently, embedded vision platforms have become prevalent for applications in video content analysis (VCA). VCA is a relatively new field in video processing. It is generally implemented using two chips, with the image signal processing (ISP) part in a digital signal processor (DSP) or field programmable gate array (FPGA) and the VCA part executed by a processor. However, a new generation of SoC FPGAs that incorporates a processor and FPGA into a single chip (Zynq) makes it possible for a single chip to perform as both an ISP and VCA. The primary constraints in an embedded vision application are power management, computational operation, and the number of utilized resources (cells). As such, these define the efficiency of the designed system.

One of the active research areas in VCA is human activity recognition, where many researchers aim at developing an algorithm for a better rate. However, few researchers have focused on developing a hardware circuit for implementing the entire process in an embedded system to achieve a real-time system for human activity recognition. Besides, in the past, detailed comparisons of computation speed between embedded vision platforms and personal computers still have not been discussed. In addition, one of the keys to realizing the benefits of FPGA, developing a parallelism algorithm, or trying to optimize the algorithm to execute in parallel is to obtain the results of unrolling and pipelining in order to provide better comparison.

In this study, human activity recognition was successfully implemented on an embedded vision platform to reduce the system cost and size. Our proposed system combines the following submodules: foreground and background identification, human detection, and feature descriptor, together with support vector machine (SVM) classification.

We make the following contribution in this study:

- We present a unified pipeline architecture for a feature descriptor in 3D science: one that is used for recognition.
- We efficiently utilize resources to allow for parallel processing.

The rest of the paper is organized as follows: in section 2, we introduce related works that have covered both the proposed action recognition algorithms and the implementation of vision algorithms on hardware; in section 3, we describe our proposed method for human action recognition; in section 4, we present the design of modules for an embedded video-processing platform; in section 5, we discuss the results of the implementation and an evaluation; and in the last section, we present the conclusion.

II. Related Work

This section discusses related work on human action recognition and the methodologies that have been used to implement recognition algorithms in embedded systems.

A. Human Action Recognition

The algorithms that were proposed for human action recognition can be categorized as either model-[1] or appearance-based. Since a large degree of variability exists in human motion, model-based methods are extremely challenging: they require high-dimensional models because of the highly articulated nature of the human body. Appearance-based methods rely on interest points that are extracted from 2D and 3D images and then employ classifiers to categorize the actions. The advantage of this approach is that it does not rely on

978-1-5090-2088-1/16/$31.00 ©2016 IEEE

human models but rather is defined on the basis of motion characteristics. Separable linear filters [2] or spatiotemporal Harris corners [3] are two early interest point-detection methods that are encoded in several ways such as SIFT, SURF, FREAK, BRIEF, ORB, pixel gradients, or jet descriptors [4], which utilize 2D data. Since depth data are lost because of projection, some of the researchers have tried to include depth data, where the two main approaches are stereovision [5] or depth camera [6][7] (Microsoft Kinect and SoftKinect). Li *et al*. [8] proposed a bag of 3D points from a depth image and modeled the posture by the Gaussian mixture model (GMM). An action graph was used for classification [9]. Lv and Nevatia [10] used the 3D information of joints with a state-based model, hidden Markov model (HMM). A similar approach was proposed by Han *et al*. [11], where a 3D joint position was described by a conditional random field (CRF). Adopting local interest point-based methods to operate in depth sequences was proposed by Hadfield *et al.* [12]. with two popular feature descriptors in 2D scenes: bag of visual words (BOWs) and relative motion descriptor (RMD), which were extended for use in 3D scenes. Other approaches tried to extract the human skeleton [13] by using a local occupancy pattern and pairwise distance features. The features were extracted from each frame, and then, a Fourier transform was used to describe the temporal variations. Linear discriminant analysis (LDA) [14] was used to select sub-volumes, and the most discriminative sub-volumes were retained. Utilizing global features is another popular method for human action recognition. In [15], the depth video is summarized in one image, which is obtained by averaging the differences between depth frames. However, this approach is only robust in applications with stationary cameras, and its detection is uninformative when a dynamic background or camera motion exists.

B. Embedded Vision System

Image and video processing applications mostly require considerable data processing and sometimes complex mathematical operations, requiring a high-performance CPU or even a multi-core system. However, the new portable vision applications, which have requirements of power efficiency, cost, and stability, provide an embedded system solution to the designer. At present, the options for implementing a computer vision algorithm on hardware include DSPs, FPGAs, mobile PC processors, and SoC FPGAs. Of course, each of these implementations is highly application-dependent. However, most of the embedded vision applications are classified as follows:

- **Image acquisition:** whereby a digital 2D/3D image is captured by one or multiple image sensors.
- **Preprocessing:** whereby basic enhancement techniques such as noise reduction and lens distortion are employed in pre-processing.
- **Feature extraction:** whereby the interest points such as edges, corners, and blobs are detected.
- **Segmentation:** is an option that depends on the type of application.

Figure 1: Image (left), probabilty Image (middle) and human detection result (right)

- **High-level processing:** which generates the final results that could be the result of the recognition algorithm.

Other studies [16] [17] [18] [19] [20] discuss the FPGA implementation of GMM, whereas [21][22] discuss the results of implementation on a graphics processing unit (GPU). A human detection algorithm is discussed in [23][24]. Real-time video segmentation is covered by [25]. Feature-based processing with SoC FPGA (Zynq) is discussed in [26], and finally, [27] discusses gender and age classification with an FPGA development system.

III. ALGORITHM

The proposed system consists of two preprocessing steps: foreground probability, which identifies the pixels that belong to the foreground, and a second preprocessing step, which identifies humans in the scene. These two preprocessing steps remove unimportant parts of the scene, which improves the results of interest point detection. Next, a computing feature descriptor, histogram of oriented 4D surface normal (HON4D), is used in the region of interest (ROI) of human detection, and finally, SVM is used to categorize the action.

A. Foreground Probability

As discussed earlier, in order to reduce the computation process, one solution is to filter out the unnecessary parts. This preprocessing takes advantage of local texture features and depth data that are represented by local binary patterns (LBPs) and photometric invariant color measurements in the RGB space. Details of the mathematical operations and hardware implementation have been previously presented [28]. The output of this module gives the foreground probability (Figure 1) for each point, which can be used in the next step to determine the ROI.

B. Human Detection

The results of foreground and background identification are determined by the ROI in complex streams that consist of humans with other objects; therefore, we need more filtering to reduce the ROI to the area that contains humans. Since this research focuses on a real-time system, the detection algorithm should be computationally efficient and require few resources. The extended algorithm in this section is based on extending a previously proposed method that used features extracted from the Riemannian manifold of the region covariance matrices computed from 2D data [29]. The proposed method considers both 2D and depth data (3D). The LogitBoost classifier is employed to detect humans. The details of the implementation and algorithm are presented in [30].

C. Histogram of Oriented 4D Surface Normal

The results of the two above mentioned preprocessing steps used to identify the ROI can also be used to extract the pixels that have rich information. The next step is to compute the 4D surface normal [31]. Consider pixel x in depth image D in a 3D space. We can extract each pixel depth at location $z = I_{Depth}(x, y, t)$. Let n represent the normal to the surface S, which is defined as

$$n = \nabla S = (\nabla D_x, \nabla D_y, \nabla D_z, -1)^T \quad (1)$$

$$\nabla D^t_x(x, y) = I^t_{Depth}(x + d, y) - I^t_{Depth}(x - d, y) \quad (2)$$

$$\nabla D^t_y(x, y) = I^t_{Depth}(x, y + d) - I^t_{Depth}(x, y - d) \quad (3)$$

$$\nabla D^t_t(x, y) = I^t_{Depth}(x, y) - I^{t+1}_{Depth}(x, y) \quad (4)$$

where d is the shifting position in the x- and y-directions. By normalizing the computed normal, the unit length normal $\hat{n}$ is obtained. It has richer information, which makes the corresponding distribution over the bin significantly different. It is included with the magnitude of gradient, which can be used to select bins:

$$m(x, y, t) = \sqrt{\nabla D_x(x, y, t) + \nabla D_y(x, y, t) + \nabla D_t(x, y, t)} \quad (5)$$

In order to compute the histogram of the normal, it is necessary to quantize the corresponding space into specific bins. Traditionally, in histogram-oriented gradient, it is obtained by quantizing a circle in order to obtain the bins of the histogram, whereas in [32], the resulting predefined orientation vector is used to find the corresponding response per direction. In this research, the 4D space is quantized by utilizing polychorons [33] [34]. A polychoron is a 4D extension of a 2D polygon, and divides the 4D space uniformly with its vertices. In this work, we used polychorons with 600 cells $\{3, 3, 5\}$(Schlafti symbol). They were constructed using the golden ratio $\phi = \frac{2}{(1+\sqrt{5})}$ on the edge of an octahedron. All sides were composed of tetrahedrons, where 20 sides meet at each vertex. According to [33], the origin of 4D space can be obtained according to the edge length $(\frac{1}{2}, 1, \frac{1}{\phi})$. By permutation, this is done as follows: 8 vertices of the form $(0, 0, 0, \pm 1)$, 16 vertices of the form $(\pm\frac{1}{2}, \pm\frac{1}{2}, \pm\frac{1}{2}, \pm\frac{1}{2})$, and 96 vertices $(\pm 1, \pm\phi, \pm\frac{1}{\phi}, 0)$. The 120 vertices quantized the 4D space, where each vertex vector was considered as a projector that was used to obtain the component of each normal from the inner product of a corresponding projector. As a result, the distribution of 4D normal orientation can be obtained as follows:

$$P(p_i|\eta) = \frac{\sum_{j\in\eta} \langle \hat{n_j}, p_i \rangle}{\sum_{p_v \in P} \sum_{j\in\eta} \langle \hat{n_j}, p_v \rangle} \quad (6)$$

where P and η are the sets of projectors and unit normal, respectively. The depth sequence is divided into $w \times h \times t$ spatiotemporal cells and is employed for each HON4D cell, and the result of each cell is collected to generate the final descriptor.

D. Classification

The classification was performed using a multi-class SVM with a kernel radial basis function (RBF). The decision function and kernel are given as follows:

$$sign(\omega^T \phi(x) + b) = sign(\sum_{i=1}^{\iota} y_i \alpha_i K(x_i, x) + b) \quad (7)$$

$$K(x_i, x_j) = exp(-\gamma \|x_i - x_j\|^2) \quad (8)$$

The weight vector, w, and the bias, b, are determined during the training phase.

IV. IMPLEMENTATION

A schematic diagram of the SoC FPGA-based human action recognition scheme for 3D natural scenes is shown in Figure 2. The architecture of the SoC FPGA provides an excellent acceleration platform because of the concurrent data processing that can be achieved by a programmable logic (PL) block and the sequential processing that can be executed with a system processor (PS), both of which are fabricated into a single chip. The main challenge facing hardware implementation is to be able to take advantage of these features and provide the data from memory when needed at the required bandwidth. Figure 2 illustrates the generic SoC FPGA-embedded vision architecture used in this study. It includes a dual Advanced RISC Machines (ARM) processor, video input, video preprocessing, video processing, and video output, and is connected to the HDMI driver. The implementation consists of three preprocessing submodules:

- **Preprocessing 1:** Captures an input stream from a camera or storage memory. In this study, all of the videos stored in the SD card are transferred by the first processor (PS1) to the two blocks of the VDMA to store RGB and depth data. The value of luminance for each pixel is determined by the circuit implemented based on equation 9. The luminance values are stored in 8-bit blocks in the RAM.

$$y = \left[\left[\begin{bmatrix} R & G & B \end{bmatrix} \times \begin{bmatrix} 66 \\ 129 \\ 25 \end{bmatrix} + 128 \right] >> 8 \right] + 16 \quad (9)$$

- **Preprocessing 2:** As discussed earlier, to filter out unimportant changes and objects and distinguish foreground from background, we developed a foreground and background identification scheme. The details of the hardware implementation are explained in [28].
- **Preprocessing 3:** The next preprocessing step is used to further filter out objects in the scene to detect humans. The mathematical operations and implementation are discussed in [30].

The results of these three preprocesses generated a mask image that can be used in the depth image to identify the scenes that include humans. The advantages of these three processes are more useful in complex streams such as Hollywood datasets. The next step is computing the HON4D descriptors, where algorithms 1 and 2 summarize the parts of sequential operation for computing the normals and histogram that could be optimized for parallel processing. In order to optimize the

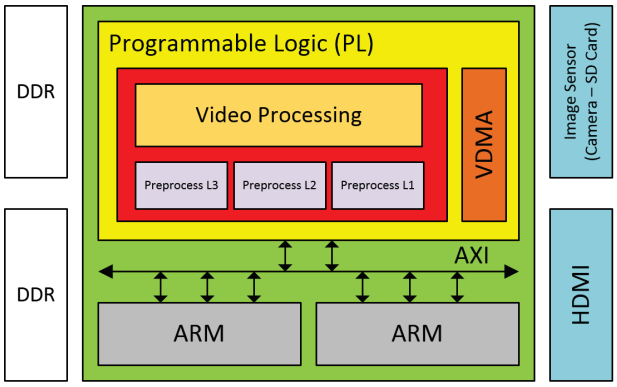

Figure 2: SoC FPGA block diagram showing video devices in the FPGA fabric that accelerates vision algorithms.

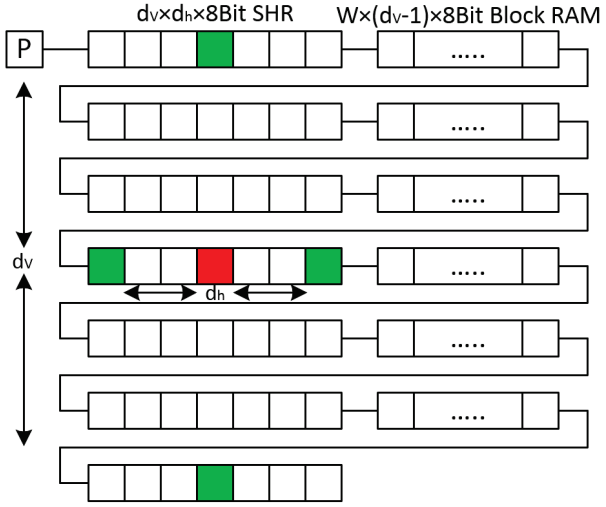

Figure 3: The algorithm 1 computation circuit.

algorithm and parallelize the operation, the types of loops need to be studies. By identifying an unrolled loop, we can improve the utilization of massive computation resources in the FPGA. The types of the loops can be classified as:

- **Irrelevant:** If the loop index is not used in any function of array I, it is considered irrelevant to array I.
- **Independent:** If the union of data spaced on array A is completely separable along the loop index, it is called a loop index independent of array I.
- **Dependent:** If the union of data spaced on array A is not separable along the loop index, it is called a loop index dependent on array I.

Considering algorithm 1, which consists of two loops, all of the arrays are dependent on the loop iterators i and j. However, the results of each operation that are independent of each other can be processed in parallel. The appropriate circuit for algorithm 1 is illustrated in Figure 3, which consists of a $d_v \times d_h \times$8-bit shift register that is accessible by each cell in parallel and a $w \times (d_v - 1) \times$8-bit block of RAM, where d_v and d_h are the vertical and horizontal step sizes. The interested pixels values are passed to the arithmetic logic unit (ALU) to

Algorithm 1 Compute d_x,d_y and d_t in the depth image

function GRADIENT($I^t_{depth}, I^{t+1}_{depth}$)
  for $i \leftarrow 1$ to h **do**
    for $j \leftarrow 1$ to w **do**
      $d_x(i,j) \leftarrow I^t_{depth}(i+d,j) - I^t_{depth}(i-d,y)$
      $d_y(i,j) \leftarrow I^t_{depth}(i,j+d) - I^t_{depth}(i,j-d)$
      $d_t(i,j) \leftarrow I^t_{depth}(i,j) - I^{t+1}_{depth}(i,j)$
    end for
  end for
  return d_x, d_y, d_t
end function

compute d_x and d_y. To compute d_t, the depth value of the next frame is needed, which is accessible from the block of the RAM with size $w \times h \times 8$.

The result of the square root is extracted by an arithmetic unit, and in order to compute the normalized n, the obtained normal is stored in a buffer to compute the normalized value. The result of the histogram is obtained by algorithm 2, which consists of six loops: the first three determine the location of cells, and the remaining three define the positions of the interested pixels in a cell where a value for the gradient is needed. Since the sizes of the cells remain constant, the values of x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, and z_{max} can be precomputed for each cell, Furthermore, because the operation of each cell is independent of the others, the operation of each cell can be executed in parallel.After the optimization of the three outer loops, the range of the three inner loops can be constant, and pipelining optimization can be employed. The last module of implementation is classification, and in this study, SVM is used. SVM takes a vector as an input and returns the class of the vector as the output. It works with a few steps, which are repeated for each support vector. Algorithm 3 describes these steps. According to the algorithm, most of the time is consumed by the main loop as it iterates many times while performing complicated operations such as multiplication and exponential operations. To parallelize the operation, the shared data S are duplicated, and at the end of the operation, the duplicated values of S are merged. The second loop, defined on the basis of the number of classes, can be run in parallel. The results of the summation and comparison can also run in parallel by defying the duplicate of the vote counter, which is needed to merge at the end. The SVM classifier is optimized as above for parallel operation, except for the main loop. The weight vector w_i and the bias b (Equation7) are constant for all evaluations loaded by the processor.

V. IMPLEMENTATION RESULTS AND EVALUATION

The proposed hardware acceleration for human activity recognition was implemented on a Xilinx XC7Z020 board equipped with an XC7Z020-1CLG484C device with an FPGA clock running at 140 MHz and an ARM clock running at 866 MHz. The circuit was synthesized on the FPGA using Vivado

Algorithm 2 Compute the histogram of each cell

```
function HISTOGRAM(D, R, C, d_x[], d_y[], d_z[])
    for d ← 1 to D do
        for r ← 1 to R do
            for c ← 1 to C do
                x_min ← cell_width × c
                x_max ← x_min + cell_width × c
                y_min ← cell_height × r
                y_max ← y_min + cell_height × r
                t_min ← cell_height × r
                t_max ← y_min + cell_height × d
                for t ← t_min to t_max do
                    for x ← x_min to x_max do
                        for y ← y_min to y_max do
                            d_x ← d_x[x] = {...}
                            d_y ← d_y[y] = {...}
                            d_t ← d_t[t] = {...}
                            h ← H(d_x, d_y, d_t) + h
                        end for
                    end for
                end for
            end for
        end for
    end for
    return h
end function
```

Algorithm 3 Support Vector Machine (SVM)

Require: $\overline{V}$ vector to classify

```
function SVM(V̄)
    for x_i ← 1 to V̄ do
        c_i ← the class of x
        k_i ← the class of K(x_i, x)
        for All c class do
            if c_i ≠ c_j then
                d ← index of c_i and c_j
                S_d ← S_d + y_{d,i} × α_{d,i} × k_i
            end if
        end for
    end for
    for All decision d do
        if S_d − b_d > 0 then
            V_i ← V_i + 1
        else
            V_j ← V_j + 1
        end if
    end for
    return c_n with the highest V_n
end function
```

2015.2. The results of our implementation are summarized in Table I. We tested the proposed system using a publicly available action recognition database of unconstrained stereoscopic 3D videos obtained from Hollywood movies [12] and MSR Action 3D datasets [34]. Our embedded human action recognition system performed at 12 fps and provided a mean precision [35] of 19.6% for Hollywood datasets and 75% for MSR Action datasets. The results of a performance analysis

TABLE I: FPGA Consumed Resources

Zynq 7020	Design	Available	Utilization
BRAM 18K	109	140	78%
DSP48E	121	220	55%
FF	48944	106400	46%
LUT	38304	53200	72%

between the proposed embedded system and a software-based system are summarized in Table II. The software implementations are realized in 1-thread and 16-thread using Visual Studio 2013 with Boost 1.53.0. Overall, our SoC FPGA-based implementation achieves a speedup of up to 24.05x over a software implementation of 1 thread. It also achieves a 5.29x speedup over a software implementation of 16 threads.

TABLE II: Performance comparison to CPU (3.40 GHz) in ms

Function	1thd	16thd	**FPGA**
Foreground Probability	40	14	**1.91**
Human Detection	77	24.30	**3.79**
HON4D	547	103	**21**
Total	664	141.30	**26.7**

VI. Conclusion

In this paper, an SoC FPGA-based video processing system is presented. The proposed algorithm employs parallel processing and pipeline architecture to perform real-time human action recognition. The experimental results for all tested datasets indicate that the proposed method improves average precision over conventional methods in both recognition and speed. Future work should focus on improving the hardware architecture, involve both ARM processors, and study power consumption. Also, polychrons with more cells should be used to improve the recognition results.

Acknowledgment

This study was supported in part by the Canada Research Chair Program, AUTO21, Networks of Centers of Excellence, the Natural Sciences and Engineering Research Council of Canada, and the Xilinx University Program.

References

[1] B. Farnell, "Moving bodies, acting selves," *Annual Review of Anthropology*, pp. 341–373, 1999.

[2] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*. IEEE, 2005, pp. 65–72.

[3] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[4] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local svm approach," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 3. IEEE, 2004, pp. 32–36.

[5] A. Safaei and M. Jahed, "3d hand motion evaluation using hmm," *Journal of Electrical and Computer Engineering Innovations*, vol. 1, no. 1, pp. 11–18, 2013.

[6] A. Safaei and Q. M. J. Wu, "Rehabilitation system in 3d natural scenes," in *EMERGING 2015, The Seventh International Conference on Emerging Networks and Systems Intelligence*. IARIA, 2015, pp. 42–43.

[7] ——, "Evaluating 3d hand motion with a softkinetic camera," in *Multimedia Big Data (BigMM), 2015 IEEE International Conference on*, April 2015, pp. 290–291.

[8] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3d points," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*. IEEE, 2010, pp. 9–14.

[9] ——, "Expandable data-driven graphical modeling of human actions based on salient postures," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18, no. 11, pp. 1499–1510, 2008.

[10] F. Lv and R. Nevatia, "Recognition and segmentation of 3-d human action using hmm and multi-class adaboost," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 359–372.

[11] L. Han, X. Wu, W. Liang, G. Hou, and Y. Jia, "Discriminative human action recognition in the learned hierarchical manifold space," *Image and Vision Computing*, vol. 28, no. 5, pp. 836–849, 2010.

[12] S. Hadfield and R. Bowden, "Hollywood 3d: Recognizing actions in 3d natural scenes," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 3398–3405.

[13] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[14] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from a single depth image," in *CVPR*. IEEE, June 2011. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=145347

[15] X. Yang, C. Zhang, and Y. Tian, "Recognizing actions using depth motion maps-based histograms of oriented gradients," in *Proceedings of the 20th ACM international conference on Multimedia*. ACM, 2012, pp. 1057–1060.

[16] H. Jiang, H. Ardö, and V. Öwall, "Hardware accelerator design for video segmentation with multi-modal background modelling," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*. IEEE, 2005, pp. 1142–1145.

[17] F. Kristensen, H. Hedberg, H. Jiang, P. Nilsson, and V. Öwall, "An embedded real-time surveillance system: Implementation and evaluation," *Journal of Signal Processing Systems*, vol. 52, no. 1, pp. 75–94, 2008.

[18] H. Jiang, H. Ardö, and V. Öwall, "A hardware architecture for real-time video segmentation utilizing memory reduction techniques," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 19, no. 2, pp. 226–236, 2009.

[19] M. Genovese, E. Napoli, and N. Petra, "Opencv compatible real time processor for background foreground identification," in *Microelectronics (ICM), 2010 International Conference on*. IEEE, 2010, pp. 467–470.

[20] M. Genovese and E. Napoli, "Fpga-based architecture for real time segmentation and denoising of hd video," *Journal of Real-Time Image Processing*, vol. 8, no. 4, pp. 389–401, 2013.

[21] P. Carr, "Gpu accelerated multimodal background subtraction," in *Digital Image Computing: Techniques and Applications*. IEEE, 2008, pp. 279–286.

[22] V. Pham, P. Vo, V. T. Hung *et al.*, "Gpu implementation of extended gaussian mixture model for background subtraction," in *Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), 2010 IEEE RIVF International Conference on*. IEEE, 2010, pp. 1–4.

[23] K. Negi, K. Dohi, Y. Shibata, and K. Oguri, "Deep pipelined one-chip fpga implementation of a real-time image-based human detection algorithm," in *Field-Programmable Technology (FPT), 2011 International Conference on*. IEEE, 2011, pp. 1–8.

[24] M. Hahnle, F. Saxen, M. Hisung, U. Brunsmann, and K. Doll, "Fpga-based real-time pedestrian detection on high-resolution images," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*. IEEE, 2013, pp. 629–635.

[25] J. Hiraiwa, E. Vargas, and S. Toral, "An fpga based embedded vision system for real-time motion segmentation," in *Proceedings of 17th International Conference on Systems, Signals and Image Processing. Brazil*, 2010.

[26] G. van der Wal, D. Zhang, I. Kandaswamy, J. Marakowitz, K. Kaighn, J. Zhang, and S. Chai, "Fpga acceleration for feature based processing applications," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2015 IEEE Conference on*. IEEE, 2015, pp. 42–47.

[27] R. Azarmehr, R. Laganiere, W.-S. Lee, C. Xu, and D. Laroche, "Real-time embedded age and gender classification in unconstrained video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 57–65.

[28] A. Safaei and Q. Wu, "A system-level design for foreground and background identification in 3d scenes," in *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, May,Accepted 2016.

[29] O. Tuzel, F. Porikli, and P. Meer, "Human detection via classification on riemannian manifolds," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

[30] A. Safaei and Q. Wu, "System-level design for human detection in 3d scenes," in *2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS)*, Feb 2016, pp. 191–194.

[31] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, June 2013, pp. 716–723.

[32] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010.

[33] H. S. M. Coxeter, *Regular polytopes*. Courier Corporation, 1973.

[34] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 716–723.

[35] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.