An Efficient Hardware Implementation of HON4D Feature Extraction for Real-time Action Recognition

Chia-Jung Hsu, Jia-Lin Chen and Liang-Gee Chen
DSP/IC Design Lab, Graduate Institute of Electronics Engineering
National Taiwan University, Taipei, Taiwan
{angela0130, jocelyn, lgchen}@video.ee.ntu.edu.tw

***Abstract*—Human activity recognition has been an important area of computer vision research. In this paper, we present real-time hardware implementation for action recognition with HON4D features, which outperform the methods relying on skeleton detectors. Our proposed circuit adopts sliding histogram, and several approximate techniques to reduce computation and speed up feature extraction. Furthermore, using sliding histogram allows continuous classification without video segmentation in advance.**

I. Introduction

Recognizing human activity is one of the important areas in computer vision. Its applications include surveillance, human activity analysis, and various system. Activity recognition algorithms from RGB video sequences have developed since 1980s. Recently, depth sensors have become ubiquitous. Compared with RGB videos, depth sequences provide more discriminative shape information which will benefit in object detection, and activity recognition.

In real surveillance system, high-speed activity recognition is required. The histogram of oriented 4D surface normals (HON4D) is a novel activity descriptor for depth sequences. Moreover, HON4D uses holistic methods, and still outperforms methods relying on skeleton detector such as [2]. In this paper, we present a pipelined architecture for HON4D feature extraction. Using several approximation techniques and sliding histogram design, our proposed circuit can be implemented with lower hardware costs while performing high throughput.

II. Histogram of Oriented 4D Surface Normals

This session gives an overview of the HON4D [1] feature extraction. HON4D describes the depth sequences using histogram capturing the distribution of the surface normal orientation in 4 dimensional space of time, depth, and spatial coordinates. The computation units of HON4D are spatial-temporal units, and then features in each spatial-temporal cells are concatenated as representation of the depth sequences. Figure 1 summarizes the steps for computing HON4D.

A. The 4D Surface Normal

In the context of depth sequences, each point satisfy $F(x, y, t, z) = f(x, y, t) - z = 0$. Therefore, the extended surface normal is computed as

$$n = \nabla F = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial t}, -1)^T. \quad (1)$$

Since the shape of the 4D surface F is represented by the orientation of the normal, we need to divide n by the magnitude of gradient $||(f_x, f_y, f_z, -1)||_2$ to obtain a normalized $\hat{n}$.

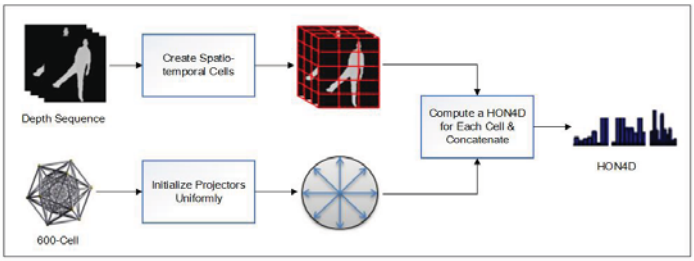

Fig. 1. The various steps for computing HON4D.

B. Histogram of 4D Normals

Quantization of the 4D space is required in order to capture the distribution of 4D normal orientation. [1] uses polychorons as projectors since a regular polychoron divides the 4D space uniformly with its vertices. In polychorons, the vertices of a 600-cell centered at the origin in the 4D space are given as:

- 8 vertices obtained by permutations of $(0, 0, 0, 1)$.
- 16 vertices obtained by permutations of $(1/2, 1/2, 1/2, 1/2)$.
- 96 vertices obtained by even permutations of $1/2(, 1, 1/\varphi, 0)$, where $1/\varphi$ is the edge length of the 600-cell, and is set to a constant called the golden ratio 2/(1 + $\sqrt{5}$).

Given the set of 120 projects $P = \{p_i\}$, and the set of unit normals $N = \{\hat{n_j}\}$ within spatial-temporal cells, we can compute the component of each normal in each direction by an inner product with the corresponding projector

$$c(\hat{n_j}, p_i) = max(0, \hat{n_j}^T p_i). \quad (2)$$

C. Normalization Computation

HON4D features are accumulation of unit surface normals within spatio-temporal cells, and normalized by the sum across all projects so that the final distribution sums to one. The normalized result can be written as

$$Pr(p_i|N) = \frac{\sum_{j \in N} c(\hat{n_j}, p_i)}{\sum_{p_v \in P} \sum_{j \in N} c(\hat{n_j}, p_i)} \quad (3)$$

Hence, a 120 dimensional HON4D descriptor describes one spatial-temporal cell. The depth sequences are divided into $w \times h \times t$ cells, and each cell is represented by a separate HON4D feature. The final descriptors is a concatenation of the HON4Ds obtained from all cells.

III. Hardware Implementation

The steps for the HON4D feature extraction include computation of normalized $\hat{n}$, histogram of projection into polychorons, and normalization. It is obvious that the calculation of HON4D feature extraction is complex and unsuitable for hardware implementation. Therefore, we propose sliding histogram and adopt approximate techniques to reduce complexity and speed up feature extraction.

978-1-4673-7365-4/15/$31.00 ©2015 IEEE

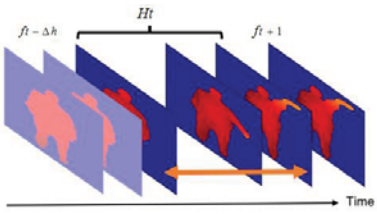

Fig. 2. The sliding histogram.

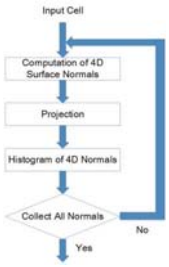

Fig. 3. Dataflow of HON4D.

A. Architecture Design

There are two major challenges in designing a real-time action recognition system with HON4D feature. Firstly, actions are continuous. One video sequence may contain multiple actions, and it's challenging to correctly tell the boundaries between two actions. Therefore, we propose sliding window technique to efficiently calculate histogram features for each sptaio-temporal cell as shown in figure 2. As a result, we can recognize different actions at any time without carefully segmentation of different actions in advance. Histogram H_t is HON4Ds between current frame F_t and previous frame $F_{t-\Delta h}$, where Δh is the length of sliding window. As time progresses, histogram H_{t+1} can be updated as

$$H_{t+1} = H_t - f_{t-\Delta h} + f_{t+1} \tag{4}$$

where f_t and f_{t+1} represent the HON4D features of frame F_t and F_{t+1} respectively. Using sliding histogram can reduce duplicate computation, and classify actions without video segmentation.

Secondly, bandwidth and memory are issues when loading entire frames to compute f_t. Since HON4D features between cells are computationally independent, data flow can be modified as figure 3. By loading each cell at once, local memory can be reduced.

B. The 4D Surface Normal

In order to obtain the normalized 4D surface normals $\hat{n}$, inverse square root operation is required. In hardware, division and square root are both computationally complex operations. Therefore, we use fast square root reduce these operations. Given the magnitude x, we want to approximate $y_{approximate} = 1/\sqrt{x}$. Firstly, magic number (0x5f3759df) is adopted to generate initial values.

$$y_{IEEE754} = (x_{IEEE754} >> 1) - magic\_number \tag{5}$$

where $x_{IEEE754}$ and $y_{IEEE754}$ are the IEEE754 presentation of values x and y, respectively. Then, $y_{IEEE754}$ is rewritten as the form of decimal representation, $y_{Decimal}$. Applying NewtonRaphson method with the initial value of $y_{Decimal}$, we can derive

$$y_{approximate} = \frac{y_{Decimal} * (3 - y_{Decimal}^2)}{2} \tag{6}$$

In our implementation, using approximate initial values as approximation of inverse square root maintains performance while saving computation. As a result, only part of fast inverse square root is implemented in our system. Figure 4 shows the architecture of fast inverse square root in our implementation, where DIC is the unit used to convert decimal representation to IEEE754 representation, IDC is the unit used to convert IEEE754 representation to decimal representation.

C. Implementation Details

Since projectors $\{p_i\}$ are predefined, we can use power of 2 to approximate vertices. Moreover, the magnitude of HON4D for spatio-temporal cells is bounded given the length of sliding window Hence, lookup table can be constructed to avoid complex operations.

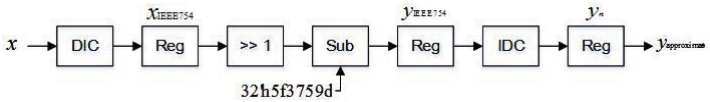

Fig. 4. Architecture of a approximate fast inverse square root.

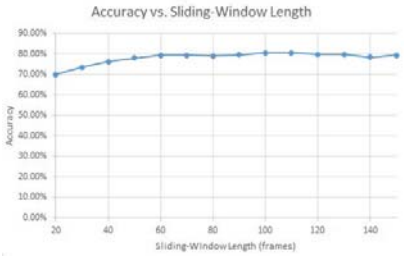

Fig. 5. Experiment results of accuracy with different sliding-window length.

IV. Experimental Results

We experimented on the proposed circuit using MSRAction3D [4] dataset. The MSRAction 3D contains 20 actions performed by 10 subjects facing a depth sensor, Kinect. The 20 actions covers various movements related to arms, legs, and torso. Therefore, this dataset is challenging as many actions appear similar. In order to determine size of sliding window, we create periodic depth sequences by circulating each depth sequence. We follow the experiment setup as in [5] (first five actors for training, and the rest for testing. The experiment result is shown in figure5. When Δh is 60 frames, we obtain the accuracy 79.46%, which is close to the baseline 87.87%.

We use Verilog hardware description language to implement the VLSI architecture, and SYNOPSYS Design Vision to synthesize our design with Taiwan Semiconductor Manufacturing Companys(TSMC) 0.13-m cell library. The synthesis result shows that our design contains 846k gate counts and operates at a clock rate of 50 MHz.

V. Conclusion

In this paper, we proposed a low-cost real-time hardware implementation for HON4D feature extraction. By using sliding histogram technique, our system can classify actions as depth sequences stream in. Furthermore, several approximation methods replace complex operations while maintaining the performance. The proposed circuit can be integrated with other intelligent system, such as surveillance systems.

References

[1] Oreifej, Omar, and Zicheng Liu. "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences." *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. IEEE*, 2013.
[2] J. Wang, Z. Liu, Y. Wu, , and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*, 2012.
[3] H. S. M. Coxeter. Regular polytopes. In *3rd. ed., Dover Publications.ISBN 0-486-61480-8*, 1973.
[4] W. Li, Z. Zhang, and Z. Liu. Action Recognition based on A Bag of 3D Points. In *CVPR Workshop*, 2010.
[5] J. Wang, Z. Liu, J. Chorowski, Z. Chen, , and Y. Wu. Robust 3d action recognition with random occupancy patterns. In *ECCV*, 2012.

978-1-4673-7365-4/15/$31.00 ©2015 IEEE