

A 4.45 ms Low-latency 3D Point-cloud-based Neural Network Processor for Hand Pose Estimation in Immersive Wearable Devices

Dongseok Im, Sanghoon Kang, Donghyeon Han, Sungpill Choi, and Hoi-Jun Yoo
Dept. of EE, KAIST, 291 Daehak-ro, Yuseong-gu, Daejeon, 305-701, Republic of Korea
Email: dsim@kaist.ac.kr

Abstract

A 3D point-cloud-based neural network (PNN) processor is proposed for the low-latency hand pose estimation (HPE) system. The processor adopts the heterogeneous core architecture to accelerate both convolution layers (CLs) and sampling-grouping layers (SGLs). The proposed window-based sampling-grouping (WSG) directly samples and groups the 3D points from the streaming depth image to boost up the throughput by $\times 2.34$. Furthermore, the max pooling prediction (MPP) predicts the 64- and 128-to-1 max pooling outputs with $\times 1.31$ throughput enhancement. In addition, the tiled data based MPP (TMPP) performs the MPP with the tiled input data to hide the latency of the MPP. As a result, the processor achieves 4.45 ms latency on the HPE system.

Introduction

A natural user interface (NUI) using finger motions and hand gestures is essential for smooth interaction with virtual objects in wearable devices such as Head-mounted Display (HMD) and Augmented/Virtual Reality (AR/VR) devices. For the accurate and robust recognition of the finger motions and the hand gestures, a hand pose estimation (HPE) system is necessary to estimate 3D position of hand joints. Recently, with the help of reliable 3D depth sensors, the point-cloud-based neural network (PNN) enables the accurate and robust HPE system [1] (Fig. 1 Top). It overcomes the drawbacks of the previous processor [5] such as the susceptibility to initialization and tracking failure caused by error accumulation. However, its software implementation on GPU results in high-latency (20.5 ms) [1] that users feel unnatural responses and interactions. Therefore, its low-latency (< 5 ms) [2] implementation is required for the fast response in the NUI, but it is challenging because of not only massive computations (2.1 GOPs/inference) in the convolution layers (CLs) but also the non-parallelized sampling-grouping (SG) algorithms such as Farthest Point Sampling (FPS), Ball Query (BQ), and Surface Normal Estimation (SNE) in the SG layers (SGLs).

In this paper, a low-latency (< 5 ms) PNN processor is proposed for the fast response in the NUI. It adopts the heterogeneous core architecture to accelerate both SGLs and CLs with the following three key features, 1) window-based sampling-grouping (WSG) to accelerate SGLs in parallel and reduce the computation costs, 2) max pooling prediction (MPP) to enhance the throughput of CLs, and 3) tiled data based MPP (TMPP) to hide the latency of the MPP. Finally, the proposed processor achieves 4.45 ms latency on the HPE system.

Proposed Point-cloud-based Neural Network Processor

The proposed processor consists of the three different parallelized cores, SG cores (SGCs), Convolution cores (CCs), and MPP cores (MPPCs) (Fig. 1 Bottom). The SGCs load the 8-bit streaming depth image from the ping-pong depth buffers and compute the SGLs using the WSG. The SGCs' outputs are passed to the CCs, and the CCs perform the convolution with the 8-bit weights from the global weight memory. One of the five input memories (IOMEMs) broadcasts the input data to the four processing elements (PEs) to calculate the convolution results of the four output channels. Each PE generates the partial sum added by the eight input channels, and they are summed up with the pre-calculated partial sums from the

IOMEM. The MPPCs perform the binary convolution (BC) with the sign bit of the weights from the global binary weight memory and propagates the MPP results to the CCs.

1. The WSG accelerates the SGLs by exploiting the adjacent pixel arrangement in the depth image (Fig. 2). The thirty-two SGCs generate the thirty-two 8×8 windows in parallel, and they are slid across the depth image in the direction of streaming with the constant step size for the uniform sampling. The point in the window is stored to serial-in parallel-out shift register (SIPOReg) which can hold maximum three points. The BQ unit loads the oldest point from the SIPOReg and groups it if its distance from the center point is less than the threshold. At the same time, the SNE unit calculates the normal vector from the three points stored in the SIPOReg for the SGL0. Then, the 3D relative coordinate of the group point with its corresponding normal vector is passed to the CC. After calculation of all 64 points in the window, the SGC computes the next window while CC starts to perform the convolution. The SGC computes the SGL1 in the same fashion with the sampled image generated by the SGL0, but only the indexes of the grouped points are stored for the subsequent CL. Since the WSG computes the SGLs not within the entire point cloud but within the window, it can reduce the 97.6% of the computations and 58.6% of external memory access (EMA) with $< 0.5\%$ accuracy degradation. Moreover, the WSG enables parallel computing with the pipeline stages. As a result, the processor boosts up the throughput by $\times 2.34$ in the HPE system.

2. The proposed MPP reduces the latency in the CLs by predicting the maximum values of the max pooling layer (Fig. 3). Since the ConvNets in the PNN have the 64- and 128-to-1 max pooling layers, the most of convolution results are not propagated to the next layers. Therefore, the MPPC performs the BC and searches the K biggest candidates as the MPP results. The MPP PE in the MPPC generates the BC results through the eight BC units and they are accumulated to obtain the final BC output. The final BC output is compared with the pre-calculated BC outputs in the candidate list (CDList), and the bigger one rewrites its index and value. Each PE in the CC generates the convolution outputs corresponding to the MPP results in the CDList and applies the max pooling among them. As a result, the processor reduces the latency of the CL2s (Fig. 3 Top Right) and it achieves $\times 1.31$ throughput enhancement in the HPE system.

3. The TMPP can hide the latency of the MPP by using the tiled data (Fig. 4). The MPPC remains idle until the CC completes the CL1 computing. To solve this problem, the CL1 is divided into four tiles in spatial x -direction. The CC performs the convolution with the one of the tiles and propagates its output to the MPPC. Then, the MPPC can compute the MPP of the tiled data while the CC computes the next tile of the CL1. As a result, the TMPP hides the latency of the other three tiles of the MPP and reduces 54.2% of the overall ConvNet latency (Fig. 4 Right).

Implementation Results and Conclusions

The proposed PNN processor is fabricated in 65 nm CMOS and Fig. 5 shows the chip photograph with the visual results of the HPE system. The processor operates in a maximum 150 MHz clock frequency with 1.2 V supply voltage. The processor achieves 4.45 ms latency, which is $\times 4.61$ faster than GPU

(Table I). The processor shows an 8.24 mm HPE error in the MSRA15 dataset [6] with 21 finger joints when K is 8-8-32 in ConvNet0-ConvNet1-ConvNet2 of the TMPP (Fig. 6). A maximum energy-efficiency is 5.62 TOPS/W, which is much higher than the previous works (Table I), at 40 MHz clock frequency with 0.78 V.

References

- [1] L. Ge, et al., "Hand PointNet," CVPR, 2018.
- [2] S. Mangiante, et al., "Vr is on the edge: How to deliver 360 videos in mobile networks," ACM SIGCOMM, 2017.
- [3] J. Lee, et al., "UNPU," ISSCC, 2018.
- [4] NVIDIA, "GTX 1080," <https://www.nvidia.com/en-us/geforce/products/10series/geforce-gtx-1080>.
- [5] S. Choi, et al., "A 9.02mW CNN-Stereo-Based Real-Time 3D Hand-Gesture Recognition Processor for Smart Mobile Devices," ISSCC, 2018.
- [6] X. Sun, et al., "Cascaded Hand Pose Regression," CVPR, 2015.

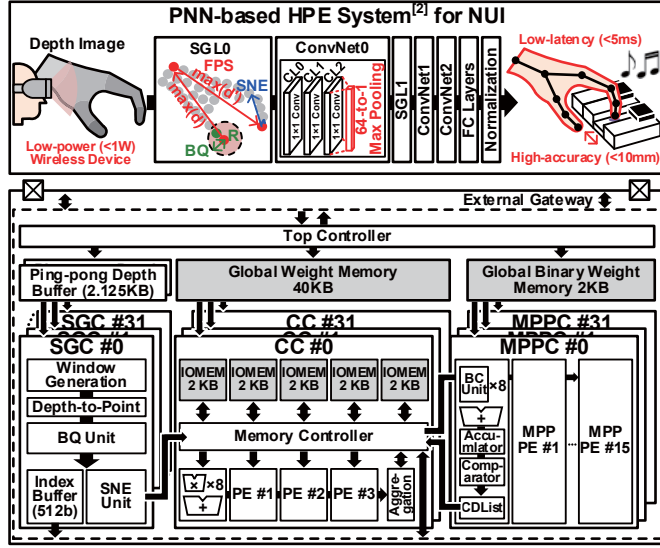


Fig. 1 HPE System Flow and the Proposed Heterogeneous Core Architecture.

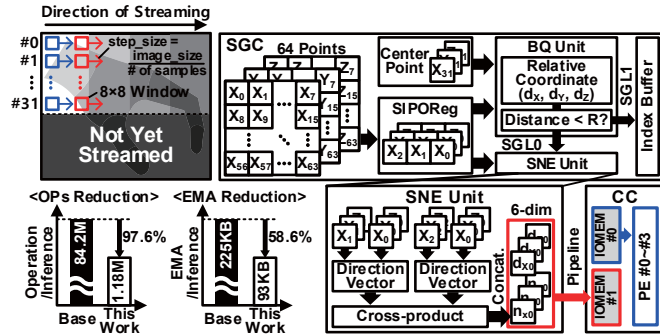


Fig. 2 SGC Architecture with the WSG Algorithm.

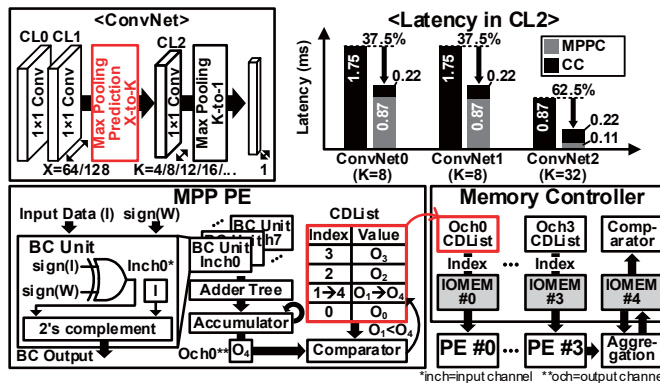


Fig. 3 MPP Concept and its Architecture for Reduction of Latency in CL2.

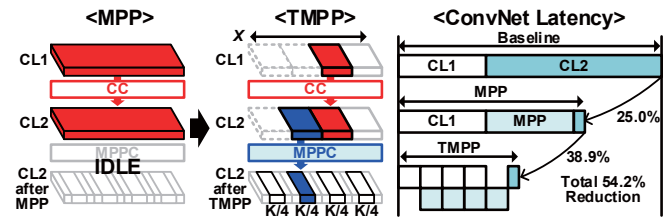
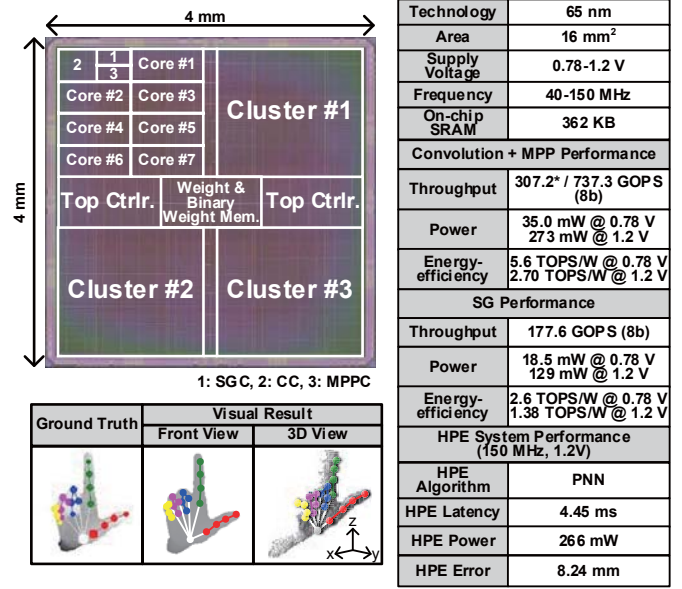


Fig. 4 TMPP Concept with Its Latency Reduction.



*Not Applying MPP

Fig. 5 Chip Photograph and Performance Summary.

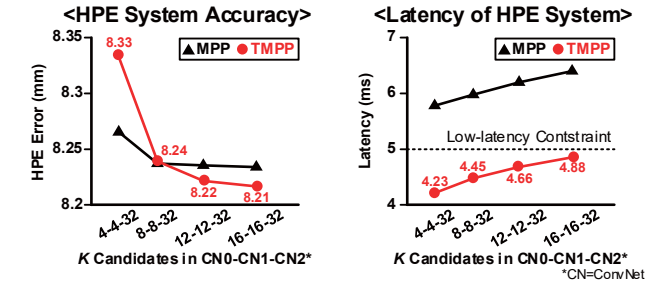


Fig. 6 Measurement Results in the HPE System.

	ISSCC18 [3]	GTX 1080 [4]	ISSCC18 [5]	This Work
Technology	65 nm	16 nm	65 nm	65 nm
Area	16 mm ²	314 mm ²	16 mm ²	16 mm ²
Supply Voltage	0.63-1.1 V	N/A	0.7-1.2 V	0.78-1.2 V
Maximum Clock Frequency	200 MHz	1,607 MHz	100 MHz	150 MHz
On-chip SRAM	256 KB	8 GB	781.5 KB	362 KB
Convolution Performance				
Peak Throughput [GOPS]	691.2 (8b)	8,228 (FP32)	19.7 (16b)	737.3 (8b)
Peak Power [mW]	@5MHz 0.63V 3.2 @20MHz 1.1V 297	180,000	@60MHz 0.9V 6.57 @100MHz 1.2V 21.3	@40MHz 0.78V 35 @150MHz 1.2V 273
Energy-efficiency [TOPS/W]	@10MHz 0.66V 5.57 @20MHz 1.1V 2.3	0.0457	@60MHz 0.9V 1.80 @100MHz 1.2V 0.92	@40MHz 0.78V 5.62 @150MHz 1.2V 2.7
HPE Performance				
HPE Algorithm	N/A	General Purpose	ICP-PSO	PNN
HPE Latency	N/A	20.5 ms ^[1]	33.3 ms	4.45 ms

Table. 1 Comparison Table.