Debris Pose Estimation by Deep Learning on FPGA

Shintaro Hashimoto (hashimoto.shintaro@jaxa.jp), Naoki Ishihama
Japan Aerospace Exploration Agency, 2-1-1, Sengen, Tsukuba, Ibaraki, 305-8505, Japan

Abstract

Although CNNs for regression problems are rarely implemented with FPGAs, our research installed debris pose estimation on an FPGA using the latest edge technology such as quantization neural network. Pose estimations were run on a workstation using 32bit floating-point precision and on an FPGA using 8bit int precision. The average errors were 4.98% and 5.38%, respectively. **This demonstrates that the regression problem can be transferred to an FPGA without a significant loss of accuracy. The FPGA power efficiency is more than 218k times that of a workstation implementation.**

Introduction

- It's difficult to install a CNN that need huge calculation for edge processing in such as satellites, automobiles, where machine resources/power are limited.
- FPGAs meet such constraints of machine resources and power associated with CNNs. FPGAs have low power consumption, but limited machine resources.
- Quantization neural networks (QNNs) have fewer parameters (bit depth) than CNNs and better estimation accuracy than Binarized Neural Networks.
- We applied QNN for the previously proposed debris pose estimation and run on FPGA. There are few examples of running regression problems on FPGA.

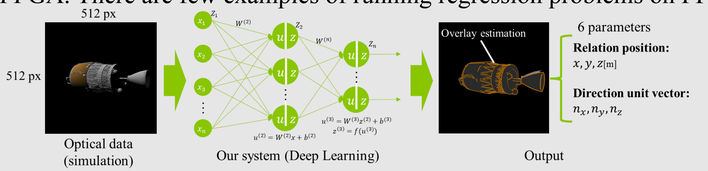

Fig. 1. Debris Pose Estimation*

*Shintaro Hashimoto, et al., "6-DoF Pose Estimation for Axisymmetric Objects Using Deep Learning with Uncertainty," 2020 IEEE Aerospace Conference, 2020.

Implementation of FPGA

Development Environment of Running CNN on FPGA

- Our environment is shown in Tab.1 and we have adopted Ultra 96 v2 as the SoC FPGA. The CLB LUT of ZU3EG was 70,560.
- Triply redundant circuits are used in satellites to prevent hardware processing errors caused by the single event setup by space radiation.
- The XQRKU060 has a proven history as an FPGA used in space, and its CLB LUT (e.g., 331k) is more than three times that of ZU3EG.

Development Procedure

- The development procedure is shown in Fig. 2. The processes that reduce FPGA performance included in CNN step (a), such as matrix decomposition, normalization, and Bayesian inference, were excluded from step (b).
- Step (c) trains and evaluates CNN. Step (d) quantizes, calibrates and evaluates CNN model of step (c) to achieve highly accurate estimation, even if low precision is used.
- Fig.3 shows the final CNN model for the FPGA. The model on the FPGA is shown as white boxes.

Tab.1. Development Environment

FPGA	SoC	Ultra 96 v2
	Chipset	Zynq UltraScale+ MPSoC ZU3EG
	CLB LUT	70,560
Workstation	OS	Ubuntu 18.04
	GPU	RTX3090
	CPU	i9-7900X
	Memory	64GB

Fig. 2. Development Procedure

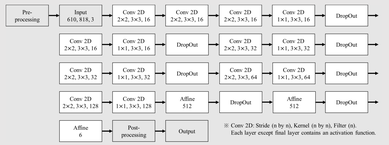

Fig. 3. CNN Model

Evaluation of Pose Estimation on FPGA

Evaluation Environment

- The pre-processing (e.g. reading and resizing images) and post-processing is processed by CPU in SoC.
- The input images are shown in Figure 4. x, y, z in design space are 28[m], 20.8[m], and 40[m] respectively. n_x, n_y, n_z is pose parameters with no rotation around the axis.

Fig. 4. Test images

Evaluation of Quantization

- Tab.2 shows a comparison of accuracy at each precision in step (d) in Fig2. The error rate in this research is the median error divided by the design space. All results were run on the workstation.
- We adopted the int 8bit parameters because of its accuracy and reduction rate of parameters. In the workstation, the power consumption in system was at least 250W. 400 images took 28.21 seconds for inference only.
- Tab.3 shows the detailed accuracy when the int 8bit CNN is operated on FPGA. The total error was 371.2, which was better than the workstation.

Tab.2. Comparison of Pose Estimation Error Rate in Quantization

Precision	x [%]	y [%]	z [%]	nx [%]	ny [%]	nz [%]	Err.
float 32bit	2.5	3.1	9.7	4.3	4.2	6.1	351.7
int 16bit	2.5	3.1	9.7	4.3	4.2	6.1	351.8
int 8bit	3.3	4.1	10.3	4.4	4.0	5.8	371.9
int 4bit	15.7	16.0	25.4	25.7	20.4	27.9	1200.1
int 2bit	16.7	16.8	26.6	28.3	19.6	49.4	1278.3

Tab.3. The Detailed Accuracy by qint 8bit CNN on FPGA

	x	y	z	n_x	n_y	n_z
Error rate	3.3 [%]	4.1 [%]	10.6 [%]	4.4 [%]	4.0 [%]	5.9 [%]
Average error	1.28 [m]	1.04 [m]	4.94 [m]	0.143	0.138	0.209
Median error	0.93 [m]	0.86 [m]	4.24 [m]	0.088	0.079	0.117

Evaluation of Performance

- Tab.4 indicates peak power in processing 400 test images on FPGA. The peak power required for this system to operate was 6.084 W. Power efficiency is more than 40 times that of a workstation implementation.
- The processing-speed of FPGA is shown in the Tab.5. Items (α) and (β) in Tab. 5 were measured at the same timing as (β) and (γ) in Tab. 4, respectively. Since item (γ) is a light process, such as normalization, there was little change in the power. Item (β-1) is the result of processing each image individually. Item (β-2) is batch processing and operated the FPGA continuously.

Tab.4. Power Consumption

State	Wat.	Amp.	Volt.
α) Standby power: OS boot only	5.796	0.483	12.00
β) CPU-processing: Pre-processing	α+0.264	0.505	12.00
γ) FPGA-processing: CNN-processing	α+0.288	0.507	12.00

Power efficiency is 12.4k [images/W] and more than 218k times that of a workstation implementation. The processing speed of FPGA was about 5.3k times faster than that of GPU.

Tab.5. Processing-speed

Processing speed	Chipset	Average time	Average deviation
(α) Pre-processing	CPU	178.08 ms	9.40 ms
(β-1) FPGA (Sequential)	FPGA	71.04 µs	3.94 µs
(β-2) FPGA (Continuous)	FPGA	13.25 µs	7.75 µs
(γ) Post-processing	CPU	8.79 ms	0.05 ms

Conclusion and Appendix

- The median errors of float 32bit and int 8bit was 4.98% and 5.38%, respectively. Since the original accuracy of CNN was lower than that of the 8bit scale, the decrease in accuracy due to quantization was at a level that was not a problem.

Appendix