Algorithm-Hardware Co-design for Deformable Convolution

Qijing Huang*, Dequan Wang*, Yizhao Gao †, Yaohui Cai ‡, Zhen Dong, Bichen Wu, Kurt Keutzer, John Wawrzynek
University of California, Berkeley, †University of Chinese Academy of Science, ‡Peking University

Motivation

- **Inefficient Model Designs** – many CV tasks use large inefficient models and operations solely optimized for accuracy
- **Limited Hardware Resources** – embedded devices have limited compute resources and a strict power budgets
- **Real-time Requirements** – accelerators must guarantee response within certain time constraints

Goals

Efficiency

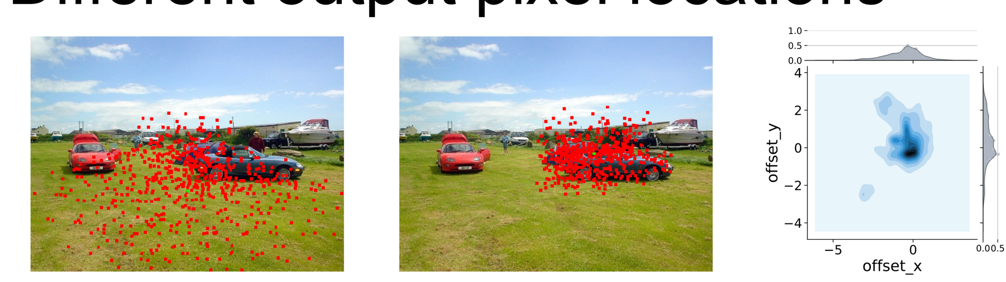
Accuracy

- Codesign **algorithms** and **accelerators** that *satisfy embedded system constraints and fall on the pareto curve of the accuracy-latency tradeoff.*

Deformable Convolution

Deformable Convolution is a *dynamic input-adaptive* operation that samples inputs from variable spatial locations

- Its sampling locations vary with:
  - Different input images
  - Different output pixel locations



a. sampling locations for lawn

b. sampling locations for car

c. distribution of the deformable offsets

Figure 1. Deformable Convolution Example

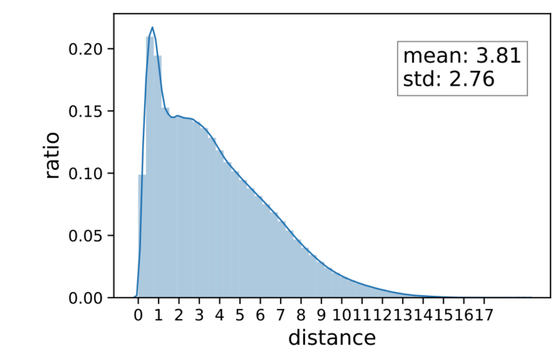
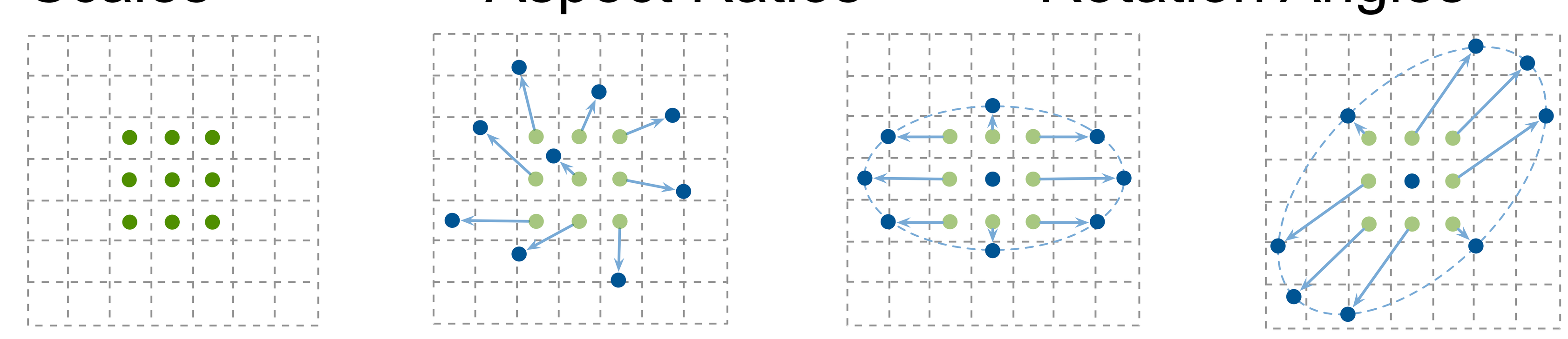


Figure 2. Distance Distribution on 5000 images from COCO

- It captures the spatial variance of objects with different:
  - Scales
  - Aspect Ratios
  - Rotation Angles



a. standard convolution (in green dots)

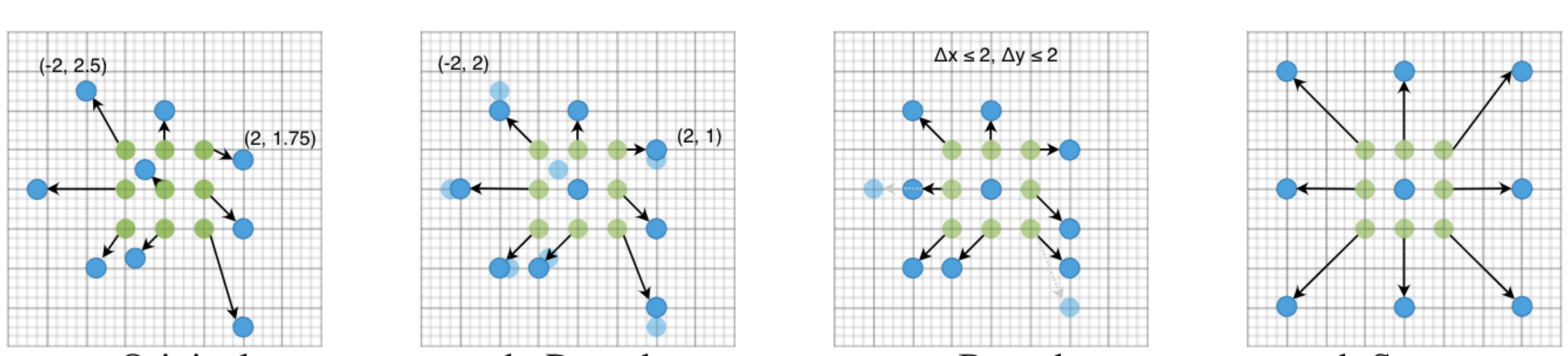
b. deformable convolution with offsets (blue arrows)

c. with different scales, aspect ratios

d. With different rotation angles

Figure 3. Deformable Convolution

Algorithm Modifications



a. Original

b. Round

c. Bound

d. Square

Figure 4. Major Algorithm Changes

a. **Deformable Convolution** samples inputs from variable offsets generated based on the input feature
b. **Rounded Offsets** rounds the fractional offsets to integer
c. **Bounded Range** restricts the range of offsets
d. **Rectangle Shape** limits the geometry to a rectangle shape
e. **Efficient Feature Extractor** uses ShuffleNetv2 as backbone
f. **Depthwise Convolution** replaces full deformable conv with 3x3 depthwise deformable conv and 1x1 conv

Hardware Optimizations

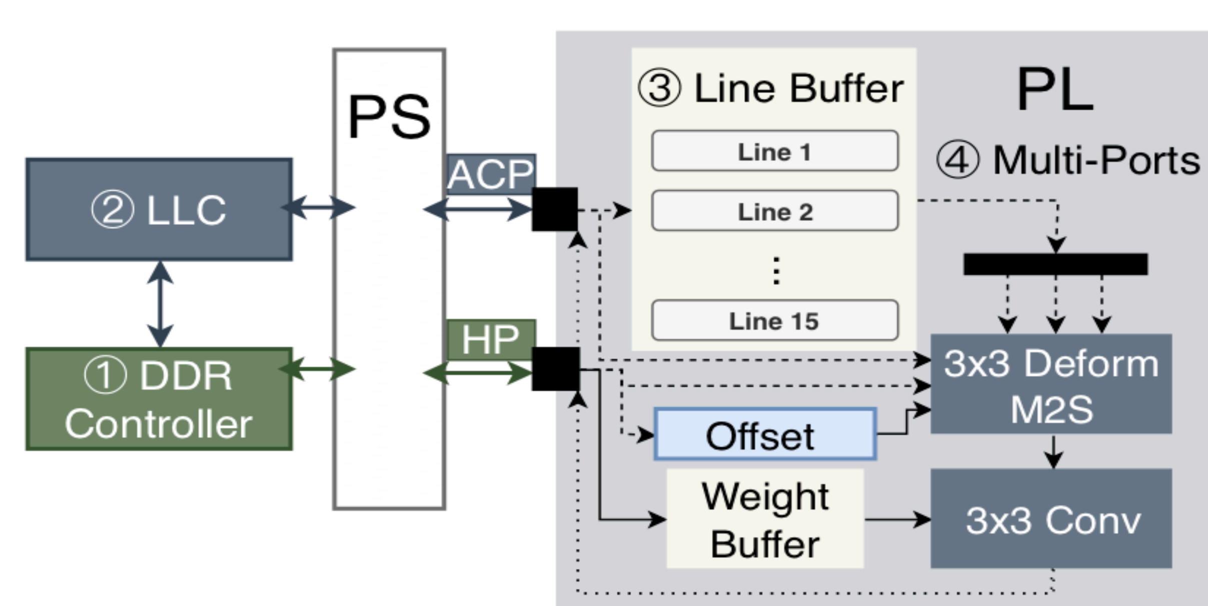


Figure 5. Hardware Engine

a. **Baseline** loads input features with dynamic offsets from DRAM directly
b. **Caching** adds LLC to leverage temporal and spatial locality
c. **Buffering** uses on-chip BRAM to buffer all inputs from limited range
d. **Parallel Ports** increases on-chip bandwidth with constrained shape

Results

Table 1. Accuracy[1] with DLA as Feature Extractor

Deformable	Round	Bound	Square	mIoU ↑
✓				79.9
✓	✓			79.6
✓	✓	✓		79.4
✓	✓	✓	✓	78.7

Table 2. Accuracy[1] with Different Feature Extractors

Feature Extractor	Operation	mIoU ↑
DLA	DeformConv	79.9
ShuffleNetV2	DeformConv	70.1
ShuffleNetV2	DeformConv + Depthwise	68.0

[1] Accuracy for Semantic Segmentation on CityScapes

- Results shows a **1.36×** and **9.76×** speedup respectively for the full and depthwise deformable conv on FPGA (Ultra96, Xilinx Zynq-MPSoC)
- **1.2** mIoU and **2.1** mIoU loss on the overall the semantic segmentation task on CityScapes respectively for the full and depthwise deformable conv

Table 3. Codesigned Hardware Performance Comparison

Operation	Original	Deformable	Bound (buffered)	Square (multi-ported)	Without LLC		With LLC	
					Latency (ms)	GOPs	Latency (ms)	GOPs
Full 3×3 Conv	✓				43.1	112.0	41.6	116.2
		✓			59.0	81.8	42.7	113.1
		✓	✓		43.4	111.5	41.8	115.5
		✓	✓	✓	43.4	111.5	41.8	115.6
Depthwise 3×3 Conv	✓				1.9	9.7	2.0	9.6
		✓			20.5	0.9	17.8	1.1
		✓	✓		3.0	6.2	3.4	5.5
		✓	✓	✓	2.1	9.2	2.3	8.2

Berkeley
UNIVERSITY OF CALIFORNIA

The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019