Porting a Convolutional Neural Network for Stereo Vision to Hardware

D. -Od. G. Sotiropoulos, Dr. M.Sc, G. - P. K. Economou,
Computer Engineering and Informatics Department, University of Patras, Greece

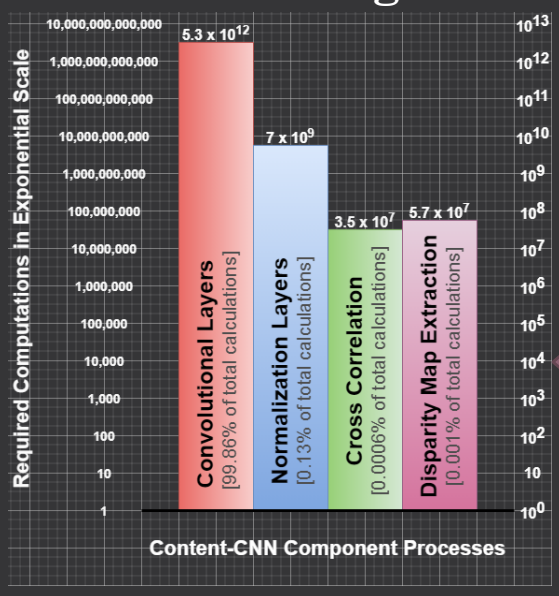
Abstract

- Popularity of CNNs leads to increasingly computationally expensive models
- Combating this, a new turn to specialized hardware demand is on the rise.
- We propose an FPGA hardware architecture, optimized towards real time execution of Conten-CNN, a novel stereo matching CNN from literature.

Content-CNN Model Quick Specifications

- 1242x375 pixel Image Input
- 2 Fully Connected CNN Paths
- 64 Neurons per Layer
- 3x3 Convolutional Filters per Neuron per Neuron Input
- Batch Normalization
- ReLU Activation Function
- Sliding Dot Product Layer

Understanding the Content-CNN Model

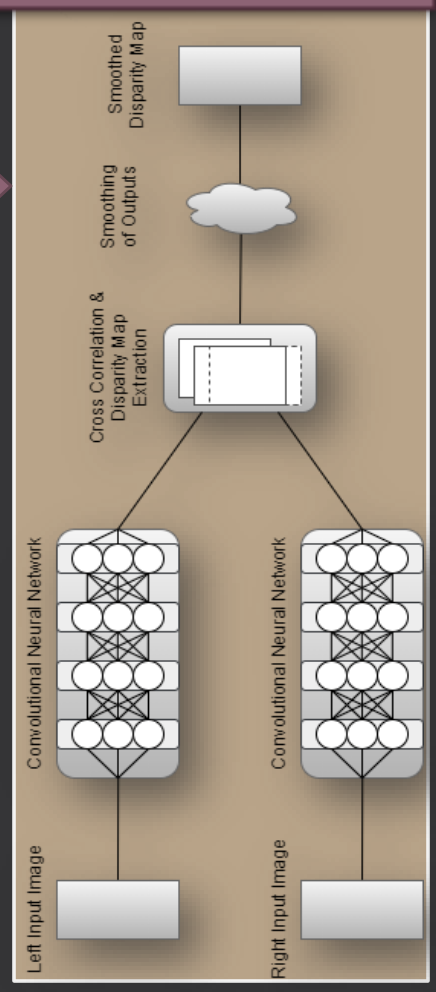


Content-CNN's [1] siamese architecture utilizes two identical CNNs to produce processed volumes from 2 input images depicting the same scene.

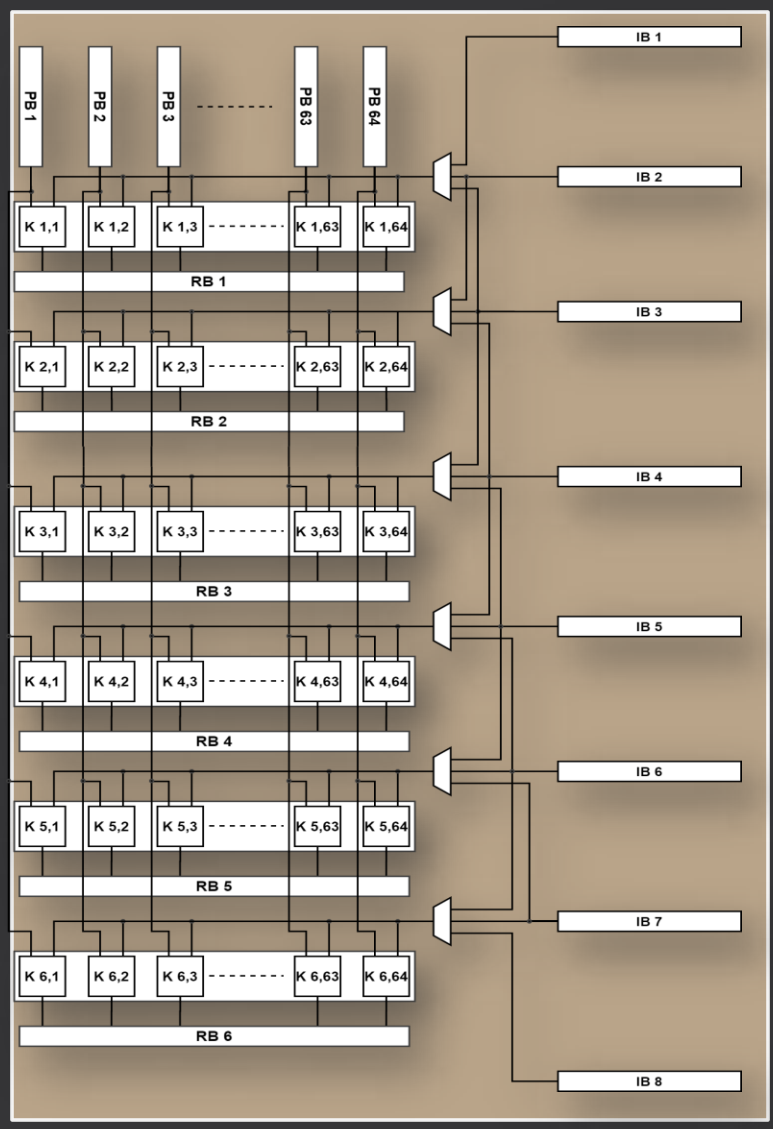
The volumes produced by the CNN are fed to a sliding dot product layer to produce a unary volume of disparities from which the disparity map is extracted.

Convolution is the basic and most computationally expensive process of any CNN. It is a sum of products derived from the application of a 3x3 filter over the whole image. Convolution takes up roughly 99.86% of the total Content-CNN calculations.

The computational cost of a single execution of the Content-CNN is approximated at **5.5 x 10^{12}** calculations. We seek to develop a system that can process this in real time.



Hardware Architecture



1. Convolutional Kernel:

We develop a convolutional kernel as the processing core of the design that can both execute the process optimally and support other functions (normalization, sliding dot product).

The kernel is serial, accumulating the sum of products of the convolution as proposed in [2].

2. Parallelization:

We propose the parallelization of neuron outputs. This approach, opens up room for memory optimization. In our final design, 6 kernel units each consisting of a kernel per neuron of a single layer (64).

Memory access is optimized this way by exploiting the accumulation of products in each convolutional kernel to accumulate the total sum of sums of products (sum of intermediate results) computing serially an output pixel per processing cycle.

3. Memory Access Optimization:

We propose systems for managing inputs that eliminate all re-reading of inputs. Additionally fit the storing of results into the processing cycle utilizing a results buffer

Pixel Buffer System

- Exploits horizontal overlap of pixels by feedback of recently used pixels that will be used in future process cycles.
- Exploits vertical overlap of pixels by feeding different kernel rows processing outputs of common inputs with pixels from the same pixel buffer.

Parameter Buffer System

- Exploits periodic application of parameters by cycling through all neuron parameters in a buffer.

Results Buffer System

- 384 results produced in parallel are serially feds to memory storage.
- Fully included in the processing cycle by exploiting long processing cycle and low memory read access.

References:

[1] W. Luo, A. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," pp. 5695–5703, 06 2016.
[2] H. Strom, "A parallel implementation of image convolution," p. 60, 2016.