Compiling Image Processing Applications for Many-Core Accelerators
Pierre Guillou

To cite this version:

Pierre Guillou. Compiling Image Processing Applications for Many-Core Accelerators. Journées de seconde année de l’Ecole Doctorale, Jun 2015, Paris, France. <hal-01178938>

HAL Id: hal-01178938
https://hal-mines-paristech.archives-ouvertes.fr/hal-01178938
Submitted on 21 Jul 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
**Image Processing**

- Mathematical Morphology Base Operators
  - Arithmetic operators
    - Unary (pixel, parameter, i input image)
    - Range (pixel, pixel, i input image)
  - Morphological operators
  - Reduction operators
  - Global max/min/sum
  - Neighbor selection, min/max/avg
  - Other operators
  - Threshold, mask, log2, ...
  - ... Sigma-C agent library

**Mathematical Morphology Image Analysis**

- Mathematical morphology: image analysis theory and technique based
- Image analysis: detect geometrical structures in an image
- Reduction operators
- Morphological operators
- Arithmetic operators
- ... Implement more complex algorithms: watershed, arrow, labelling, minima, ...

**Compilation Chain**

- Original application
- Source-to-exe compiler
- Call graph Optimizations
- Target-specific compiler
- Compute binaries

**Runtime Environment**

- Control code
- Host runtime
- Accelerator runtime on I/O clusters
- Compute clusters

**Optimisations**

- Unrolling of converging loops
- Arithmetic operators aggregation
- Generation of host-specific convolutions
- Data parallelization for compute-intensive operators

**Results: Execution Times and Energy Consumption**

- MPPA-256 (10 W)
- KALRAY (FPGA, 25 W)
- AMD 4-core (OpenCL, 90 W)
- Tesla C 2050 (OpenCL, 240 W)
- Vega 620 (OpenCL, 40 W)

**Future Work**

- Other programming models: SpiNNaker/Linux on compute clusters, communication library between clusters
- OpenCL via local memory pages
- Improve data-parallelism to better advantage of the current architecture
- Implement more complex algorithms: watershed, arrow, labelling, minima, ...

**References**