Experiences in Speeding Up Computer Vision Applications on Mobile Computing Platforms

Luna Backes, Alejandro Rico, Björn Franke

SAMOS XV, Samos, Greece
20/07/2015

THE UNIVERSITY of EDINBURGH informatics
MONT BLANC
BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Context: Computer Vision (CV)

- How to process and analyse images to **understand** them
- Societal impact
  - Education
  - Quality of life
  - Safety
- Embarrassingly parallel SPMD on large data sets

Context: Mobile devices

- Mobile devices' market growing
  - Large performance increase (25x in 3 years)
  - Affordable
  - Easy to carry anywhere
- Sensors

Problem statement

- Slow speed of state-of-the-art CV algorithms on mobile devices
  - Simplify algorithms
  - Optimise the system

Problem statement

- Slow speed of state-of-the-art CV algorithms on mobile devices
  - Simplify algorithms
  - Optimise the system

Poor user experience

Problem statement

- Slow speed of state-of-the-art CV algorithms on mobile devices
  - Simplify algorithms **Poor user experience**
  - Optimise the system ✓

↓

Software optimisation

Contributions

- Port and optimise a CV algorithm for a mobile SoC
  - Profiling
- Step-by-step optimisation and evaluation of
  - Temperature
  - Performance
  - Power

State of the art: CV on mobile

- Hardware

Project Tango

Movidius

- Software

Word lens

State of the art: CV app on GPU

- Face recognition algorithm [Cheng'11]
  - OpenGL, Android
  - Qualcomm, NVIDIA, ImgTec
  - Different algorithm, single image
- Image inpainting-based object removal algorithm [Wang'13]
  - OpenCL, Android
  - Qualcomm
  - Different algorithm, single image

Software

- KinectFusion: An algorithm used for
  - Scene scanning
  - Object recognition
  - Augmented reality
- Existing versions:
  - C++
  - CUDA
  - OpenMP
  - OpenCL

```
for all frames{
    Acquisition();          2.7 %
    Preprocessing();       23.1 %
    Tracking();             5.5 %
    Integration();         49.9 %
    Rendering();           18.6 %
    Drawing();              0.1 %
}
```

Hardware

- Arndale board: development kit with
  - 2x ARM Cortex-A15 CPUs
  - ARM Mali-T604 GPU

Benchmarks

chairs

desktop

person

weird

Measurements

- Performance: FPS
  - gprof
  - clock_gettime()
- Temperature: ºC
  - Chip sensors
- Power: W
  - Yokogawa power meter

Overall procedure

1. Refactoring
2. OpenCL porting
3. Optimisation

Optimisations

Optimisations

- Port kernels

Optimisations

- Port kernels
- Shared memory

Optimisations: shared mem

Source: ARM

Optimisations: shared mem

Source: ARM

Optimisations: shared mem

Source: ARM

Optimisations

- Port kernels
- Shared memory
- Reduce synchronisations
  - Remove OpenCL barriers
  - Remove memory mappings

Optimisations

- Port kernels
- Shared memory
- Reduce synchronisations
  - Remove OpenCL barriers
  - Remove memory mappings
- Parallelise with OpenMP

Results

1. Original version evaluation
2. Optimisations evaluation
3. Final results
4. Performance portability

Temperature

- Original version
- Thermal throttling
- Once every 20 seconds

Temperature

- Original version
- With a heat sink
- Less thermal throttling: once every 83 sec

Temperature

- Original version
- With a fan
- No thermal throttling → stable results
- Unfeasible for mobile devices

Performance

- More time at a high frequency leads to more performance

Power

12
10
8
6
4
2
0
Power (W)
chairs
desktop
person
weird
Benchmarks
Dynamic
Static
NoCooling
Heatsink
Heatsink+Fan

Execution time

Performance

Power

shared memory
OpenMP
38%
Power (W)
16
14
12
10
8
6
4
2
0
chairs
desktop
person
weird
Benchmarks
Dynamic
Static
Original
OCL-Initial
OCL-SharedMem
OCL-Raycast
OCL-Track
OCL-OpenMP
OCL-Barriers
OCL-Bilateral

Power

Performance: 300%
shared memory
OpenMP
38%
Power (W)
16
14
12
10
8
6
4
2
0
chairs
desktop
person
weird
Benchmarks
Dynamic
Static
Original
OCL-Initial
OCL-SharedMem
OCL-Raycast
OCL-Track
OCL-OpenMP
OCL-Barriers
OCL-Bilateral

Performance

Temperature

Original

Optimised

Power

10
8
6
4
2
0
Power (W)
chairs
desktop
person
weird
Benchmarks
Dynamic
Static
Original
Optimised
Same power
+
Less time
=
Energy
efficiency

Execution time

Performance

Summary

- Mobile devices are promising for CV
- CV applications can benefit from GPUs
- We found heat dissipation is a problem
- We optimised KF for a mobile SoC
  - **4x** speed up and **3x** better energy efficiency
  - 0.6-0.7 to 2.2-3.1 FPS (far from 30 FPS)
- Furthermore, our optimisations:
  - Reduced overheating
  - Are portable across SoC generations
  - Are one step towards interactive rates

Experiences in Speeding Up Computer Vision Applications on Mobile Computing Platforms

Luna Backes, Alejandro Rico, Björn Franke

SAMOS XV, Samos, Greece
20/07/2015

THE UNIVERSITY of EDINBURGH informatics
MONT BLANC
BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Refactoring

- Missing const qualifiers
- Parameters by reference
- Reduce useless function calls
- Porting C++ to C99

Context: PAMELA project

- Aims to optimise the hardware and software configs together: 3D scene understanding
- Learn about optimising many-core systems
- This work is part of the PAMELA project