GPU Computing

Projects

E. Carlinet, J. Chazalon {*firstname.lastname@lrde.epita.fr*}

April 2022

EPITA Research & Development Laboratory (LRDE)

Slides generated on April 20, 2022

Overview

Objectives

The goals of the project are to:

- apply data-parallelism concepts
- practice with CUDA
- set up a benchmark with a sound evaluation procedure
- present your results in a clear and convincing way

Possible Subjects

Standard Assignment

We propose 1 subject, that most of you should work on:

Implementation a Harris corner detector in CUDA

More details later in this presentation.

Custom Assignment

For students who are at ease with CUDA, and want to investigate a particular question:

Implementation and performance analysis of *SOME INTERESTING* **algorithm in** *YOUR PARALLEL PROGRAMMING TECHNOLOGY OF CHOICE*

If you choose this assignment, you must **validate your subject with us.**
Contact us by email ASAP.

Important dates

Deadline	What	Where
May 16, 23:59	Group composition (**teams of 4**)	on Moodle
May 16, 23:59	Final project submission (**teams of 4**)	on Moodle
May 17, all day	Oral defense	(either Teams or in presence)

Deliverables

Final Deliverables (1/3)

1. **Implementation**

- Source code for C++ CPU reference
- Source code for CUDA implementation(s)
- Source code for benchmark tools
- Build scripts (GNU Make, CMake...)

We must be able to reproduce your results

Final Deliverables (2/3)

2. **Report**

- Description of the problem
  - Detailed if custom subject
  - Quick summary otherwise
- Quick description of the baseline CPU implementation: paper reference, parallel or not, etc.
- Quick description of the baseline GPU implementation: changes from CPU version, **kernels implemented**, etc.
- **Which performance indicators you have used and why**
- **Identification of performance bottlenecks** (with measured indicators, graphs, etc.)
- **For each improvement over the GPU baseline** (implementations):
  - justification of this work regarding performance analysis
  - description of the improvement (e.g. used output privatization instead of global atomics)
  - comparison of the performance with and without this implementation
- **Table with summary** of the benchmark of all variants implemented
- **Summary of the contributions of each team member**

Final Deliverables (3/3)

3. **A live lecture / defense**

- 15’ presentation
- 5’ demo
- 5’ discussion
- Defenses will be held on Teams (opt. in presence)
- **All** the team members must be there

We will share with you some images/videos to process during the defense.

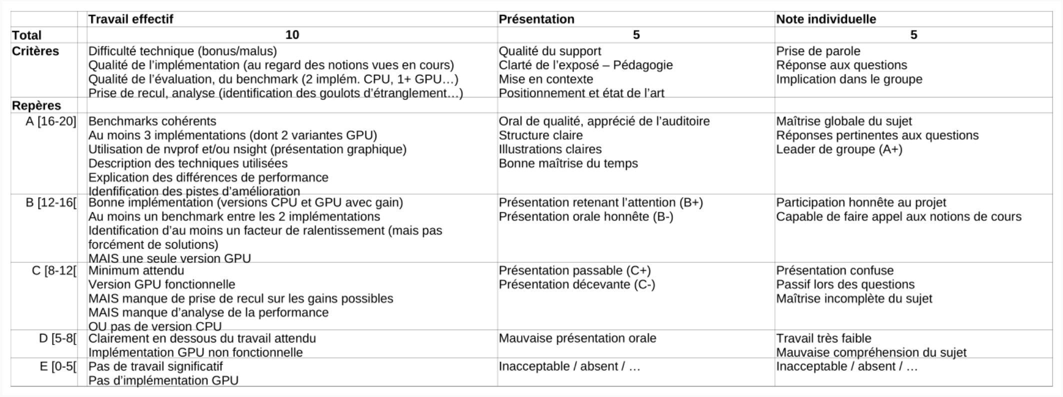
Links to each meeting will be shared when all groups are formed.

Submission

Submit code + report + slides on Moodle

https://moodle.cri.epita.fr/course/view.php?id=790

Grade Sheet Used for Last Session



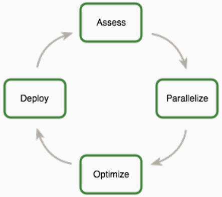
	Travail effectif	Présentation	Note individuelle
Total	**10**	**5**	**5**
Critères	Difficulté technique (bonus/malus) Qualité de l'implémentation (au regard des notions vues en cours) Qualité de l'évaluation, du benchmark (2 implém. CPU, 1+ GPU…) Prise de recul, analyse (identification des goulots d'étranglement…)	Qualité du support Clarté de l'exposé – Pédagogie Mise en contexte Positionnement et état de l'art	Prise de parole Réponse aux questions Implication dans le groupe
Repères			
A [16-20]	Benchmarks cohérents Au moins 3 implémentations (dont 2 variantes GPU) Utilisation de nvprof et/ou nsight (présentation graphique) Description des techniques utilisées Explication des différences de performance Idenfification des pistes d'amélioration	Oral de qualité, apprécié de l'auditoire Structure claire Illustrations claires Bonne maîtrise du temps	Maîtrise globale du sujet Réponses pertinentes aux questions Leader de groupe (A+)
B [12-16[	Bonne implémentation (versions CPU et GPU avec gain) Au moins un benchmark entre les 2 implémentations Identification d'au moins un facteur de ralentissement (mais pas forcément de solutions) MAIS une seule version GPU	Présentation retenant l'attention (B+) Présentation orale honnête (B-)	Participation honnête au projet Capable de faire appel aux notions de cours
C [8-12[	Minimum attendu Version GPU fonctionnelle MAIS manque de prise de recul sur les gains possibles MAIS manque d'analyse de la performance OU pas de version CPU	Présentation passable (C+) Présentation décevante (C-)	Présentation confuse Passif lors des questions Maîtrise incomplète du sujet
D [5-8[	Clairement en dessous du travail attendu Implémentation GPU non fonctionnelle	Mauvaise présentation orale	Travail très faible Mauvaise compréhension du sujet
E [0-5[	Pas de travail significatif Pas d'implémentation GPU	Inacceptable / absent / …	Inacceptable / absent / …

How you should work

Our Expectations

We expect your implementation to be:

- running on GPU;
- correct, i.e. it produces an acceptable result.



Do not try to make it fast at first, just make it work.

Then, try to apply NVidia's Assess, Parallelize, Optimize, Deploy (APOD) design cycle as described in the CUDA C++ Best Practices Guide:

1. identify the part of the code which is responsible for the bulk of the execution time;
2. get a parallel version of the code (assumed to be sequential at first);
3. optimize the performance of the parallel code;
4. measure the performance of the new code.

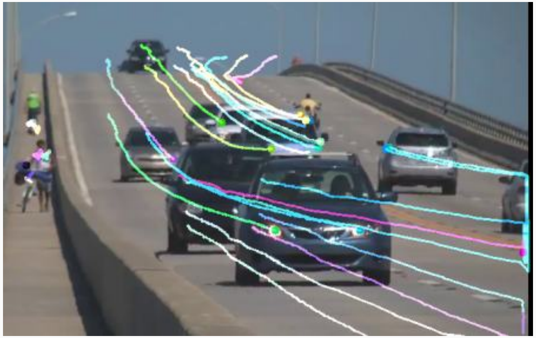
Project Outline for Standard Performance Analysis

Broad Outline	Concrete Example
Choose an application	Mandelbrot
Determine the most time-consuming part of the app	Global atomics
Determine one or more data-parallel approaches to solve the problem	Tiling...
Create multiple implementations of the approach	One naive version, one version with shared memory...
Benchmark the implementations	Record memory transfer time, kernel time, utilization, FLOPS, etc.
Relate results to course concepts	Identify the cause of the bottleneck (memory or compute bounding)

Harris corner detection with CUDA

Objective

Optical flow to track motion.



In practice: at least a working CUDA implementation which can process 1 image

Inputs and Outputs

Your program should be able to take an image as input, and return the coordinates of the 2000 best (sorted) keypoints.

You should be able to display on some image the results produced by Python (green), CPU C++ (cyan) and GPU CUDA (pink), using some tools of your choice (Python script + image viewer = perfect).

Recommended approach

1. Port the Python code from next MLRF lab session in C++ to get a working base https://www.lrde.epita.fr/~jchazalo/teaching/MLRF/202204_IMAGE_S8/practice_02_student.tar.gz
2. Port the C++ code to CUDA
3. Check all results are correct
4. Identify what can be optimized, implement, measure, repeat.

Some hints:

- Have a working (*slow*) C++ reference implementation first (and keep it forever)
- Tag (*git tag*) the versions of your program before any optimization (useful to track and benchmark ideas)
- Try optimizations step by step so that you can tell which ones are the most important

Algorithm steps

Here are the critical steps of the algorithm:

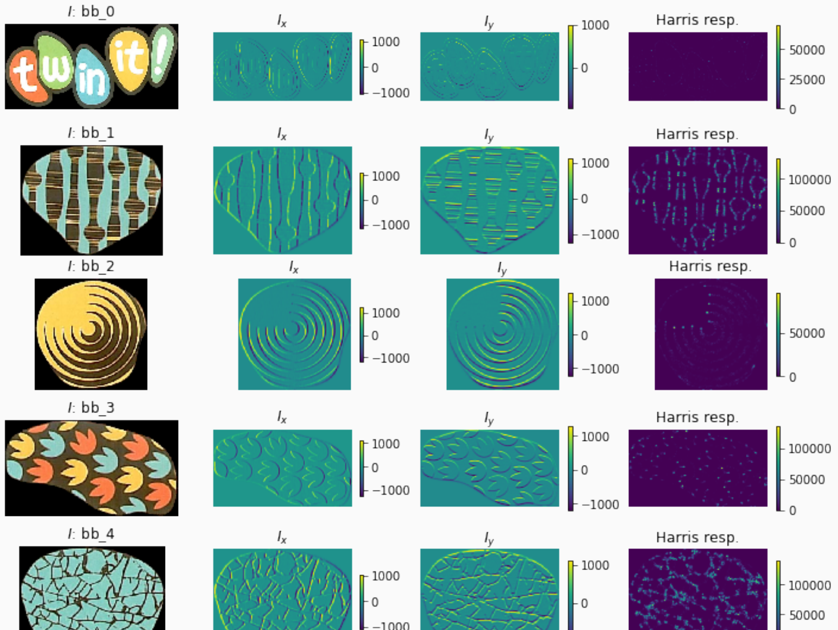
1. Copy the image to the GPU, opt. using managed memory, opt. convert colors
2. Compute the smoothed image derivatives I_x and I_y
3. Compute the structure tensor images $< I_x^2 >$, $< I_y^2 >$ and $< I_x * I_y * >$
4. Compute the approximation (or real) eigenvalue vectors and the cornerness map
5. Perform a non-maximal suppression using a morphological opening
6. Sort the filtered responses
7. Return the coordinates of the 2000 best (or less) keypoints

It is up to you to find which kernels you need, and which pattern they correspond to.

For the visualization (don't run this on the device), you need to be able to display the keypoints.

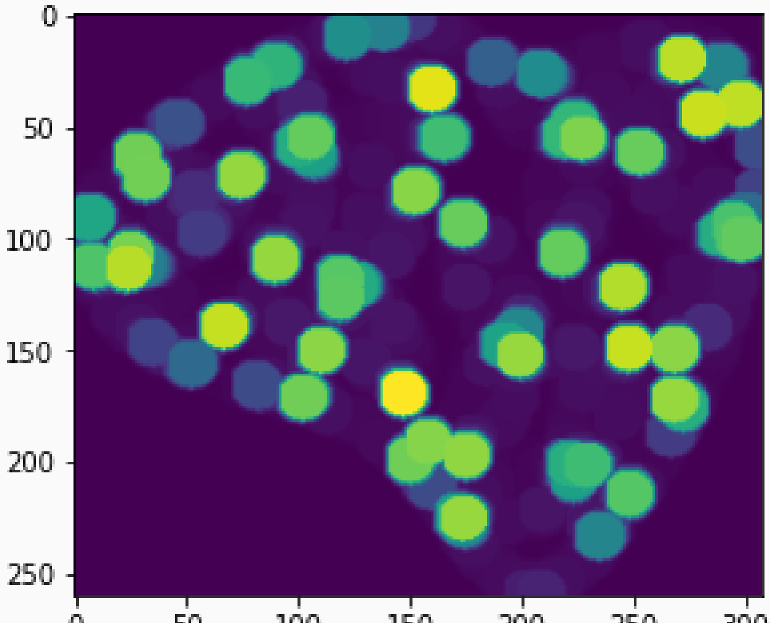
Algorithm illustrated

Image derivatives, Harris response



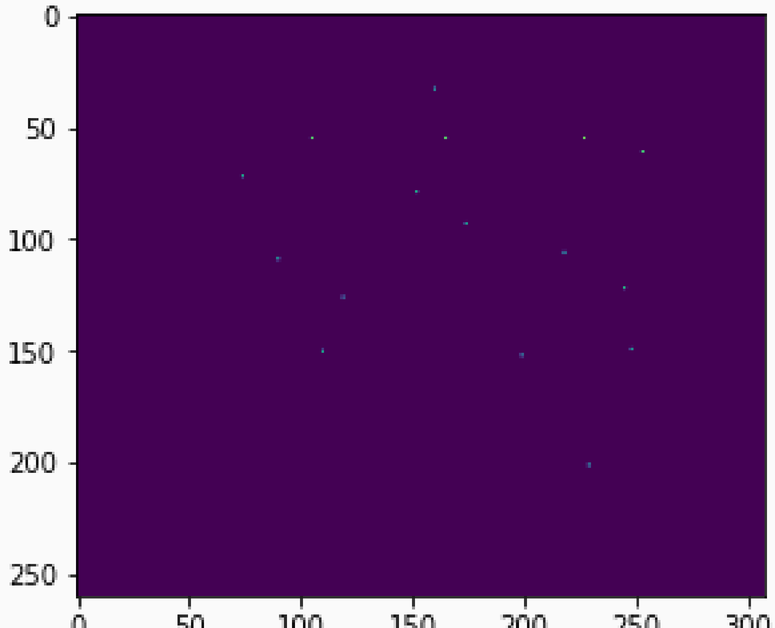
Algorithm illustrated

Opening



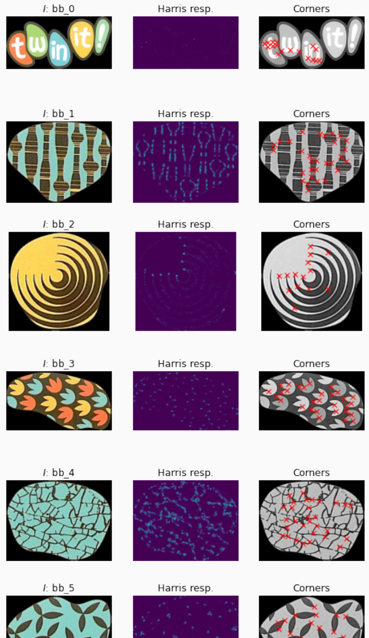
Algorithm illustrated

Maximas



Algorithm illustrated

Final detection



Expected work

To get the average grade:

- all deliverables (code, report, slides)
- working CPU version
- working GPU version
- benchmark their respective speed and compute relevant indicators (occupancy, L1/L2 cache hit rates…)

To get a better grade: - implement more variants - perform more analysis - …

Dataset

Collaborative dataset for photos and videos:

`https://cloud.lrde.epita.fr/s/HryzQFoEEdb53A7`

Each **group** have to upload 1 video and 2 photos of something with texture and simple motion:

1. 1 video (1080p or 4K, < 10s)
2. 2 photos (16-24 Mpix, JPEG)