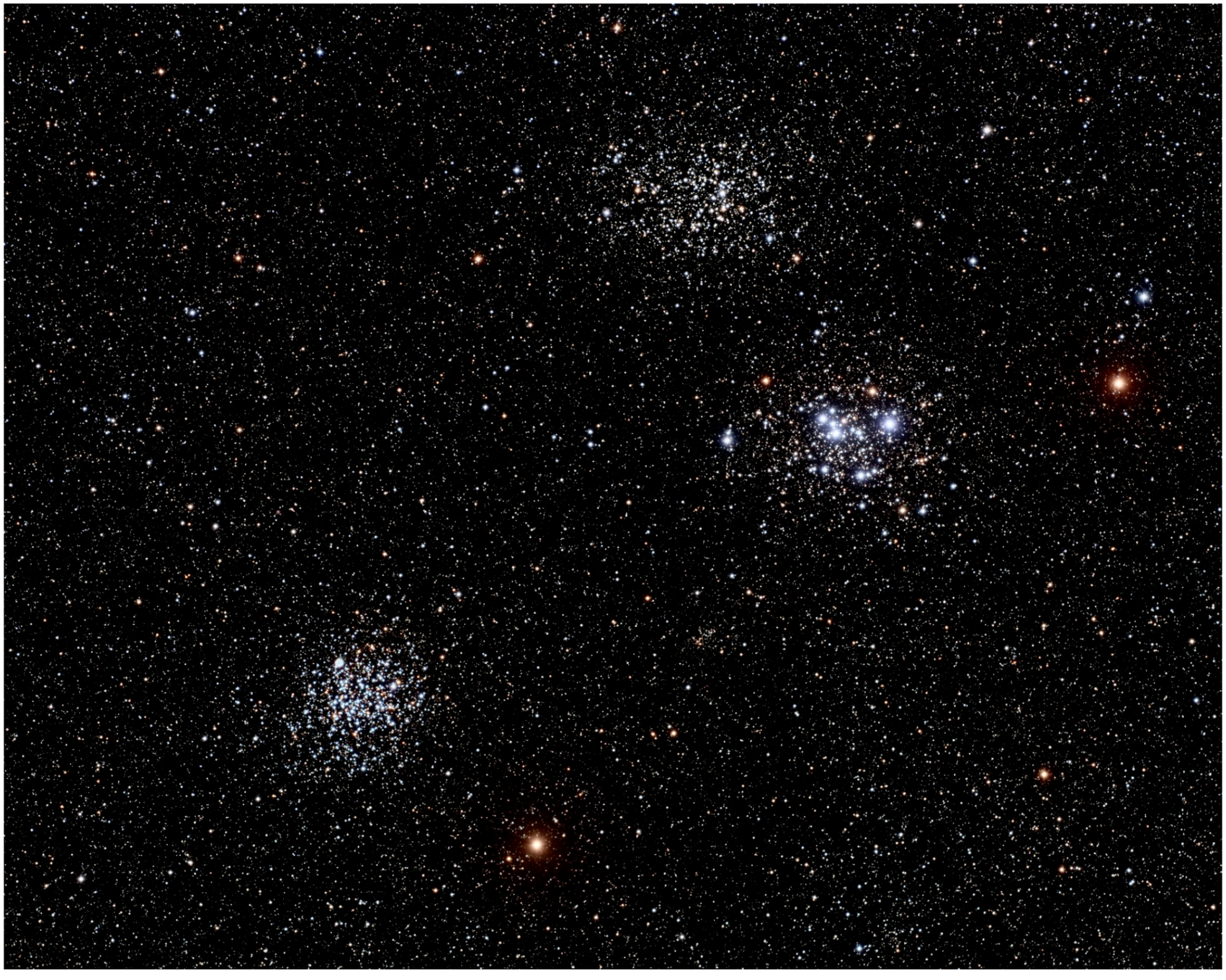
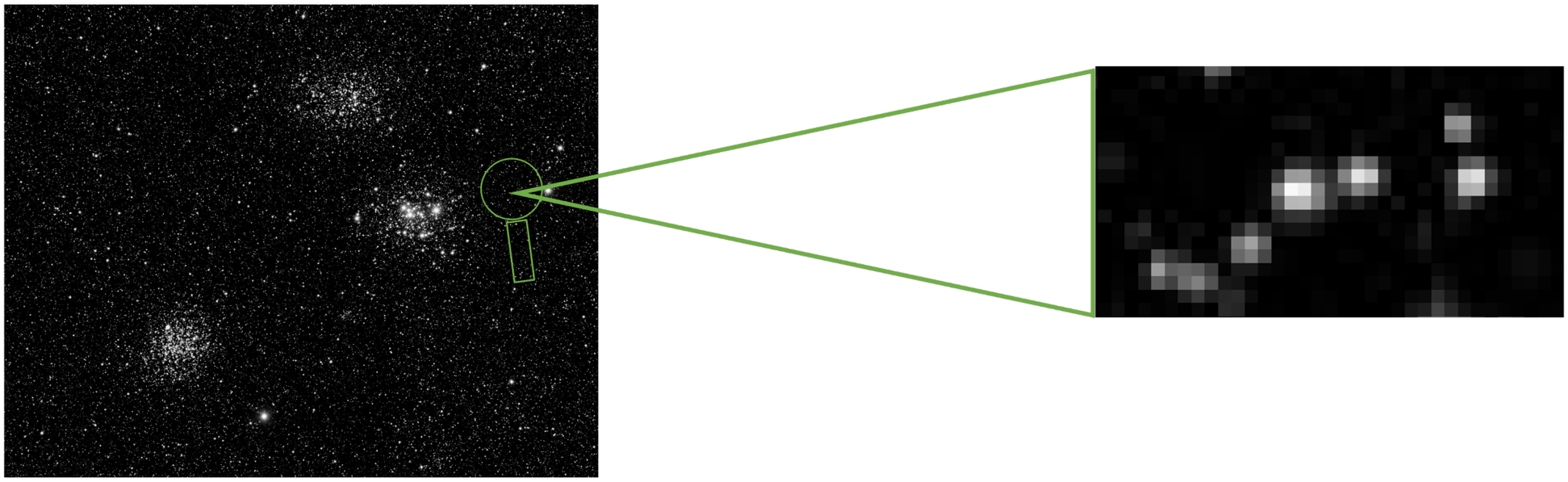
Project OpenCL: counting stars

Multicore programming

How many stars are in this picture?



Algorithm



1. Convert to grayscale
2. Compute average brightness ⇒ threshold = 2 * average
3. For each pixel:
    1. Check brightness ≥ threshold
    2. Check brightness ≥ max(neighbors) over a 7×7 window
    3. ⇒ is a star
4. Sum all matching pixels

Implementation

Sequential implementation given in Python
Your assignment: port to OpenCL & optimize

⚠️ Add edge handling technique

Many optimizations possible:

- Use of private/local memory
- Work group size
- Changing/re-ordering steps of the algorithm
- Vectorization

⇒ create several optimized versions

Always optimize for GPU first

Evaluation: correctness

Result of naive and all optimized versions should be equal to result with sequential version in Python.

You may need to change the Python version to add your edge handling technique.

Some example images are given.

Evaluation: performance

Benchmarks on sample image(s)

First: compare naive and optimized versions on GPU, with varying work group sizes

Then, at least one of these experiments:

- CPU vs GPU
- Memory transfer time
- Compare images (sizes, number of stars)
- Other algorithm parameters (threshold, window size)
- Other hardware (if you have good GPU or Apple M1/M2)
- Port to CUDA or Metal
- Others…

Evaluation: hardware

Run your experiments on:

- "Dragonfly", a machine at SOFT with a high-end GPU
  required
- Your machine / in the computer rooms
  (if the machine has a dedicated GPU or Apple M1/M2)
  optional

Requirements

<u>Either:</u>
<u>2 implementations</u>
Naive implementation
One optimization

+

<u>3 experiments</u>
Benchmark of naive + optimization
Two other benchmarks

<u>Or:</u>
<u>3 implementations</u>
Naive implementation
Two optimizations

+

<u>2 experiments</u>
Benchmark of naive + optimizations
One other benchmark

or more…

Report

Table of contents in assignment sheet:

- Implementation
  - Naive version: what does a work item / work group do?
  - Edge handling technique
  - What optimizations did you implement?
- Evaluation
  - Correctness
  - Performance

Details

Deadline: **Sunday, 14th of May**, 23:59

Submit code and report (ZIP) on Canvas

Project defense in June

⅓ of final grade

Assignment and Python implementation on Canvas