Parallel Background Subtraction with KNN

Group 13

0556055 吳雅就
0656017 顏義洋
0656132 巫謹佑

Outline

- Introduction
- Problem statement
- Environment setting
- Proposed solution
- Evaluation
- Final result
- Conclusion
- Contribution

Introduction

- Background subtraction
  - Known as foreground detection.
  - A commonly used approach for detecting moving objects in videos from static cameras.
  - The rationale approach is that of detecting the moving objects from the different between the current frame and a reference frame.

OpenCV -

- MOG2
  - serial
  - multi-thread
  - OpenCL
  - CUDA
- KNN
  - Only serial

MOG2

KNN

So we decided to parallelize it!

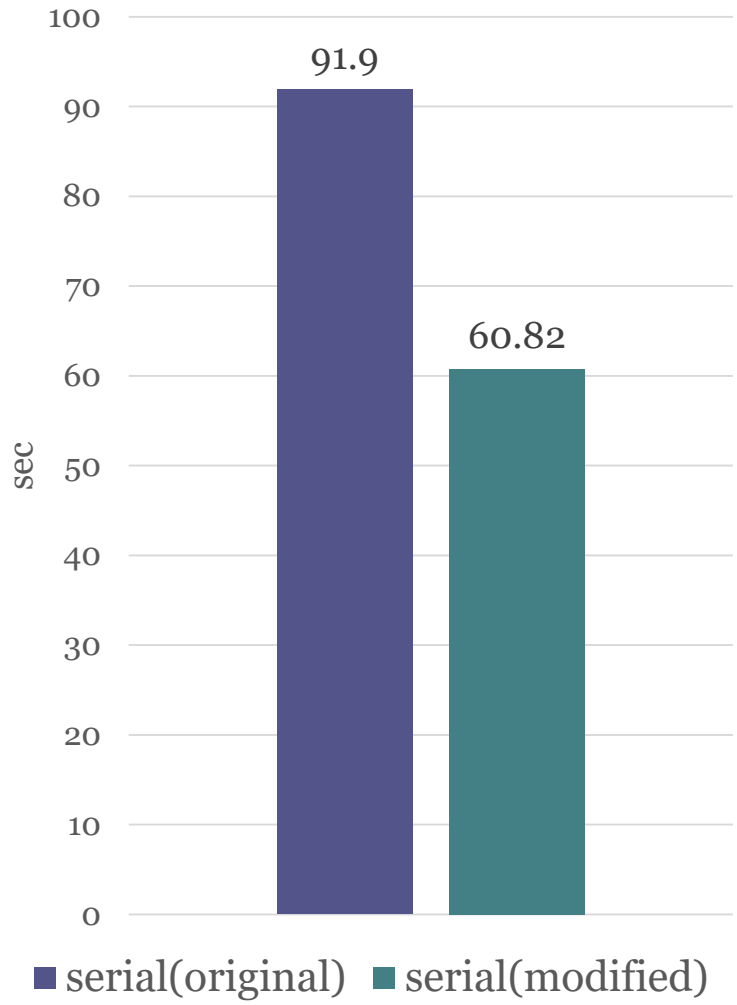
Problem statement

- KNN background subtraction
  - Pixel by pixel model
  - It randomly samples pixels in a constant interval
  - Use rand() in each model to randomly sample pixel
    - Hard to parallelize
    - Need to improve algorithm first

* Zivkovic, Z., van der Heijden, F. 2006. Efficient adaptive density estimation per image pixel for the task of background subtraction. In Pattern Recognition Letters, 27(7):773–780, 2006

After improving

- The pixel model is independent to each other
  - Use the same random parameters to all models.
  - Easy to parallelize.



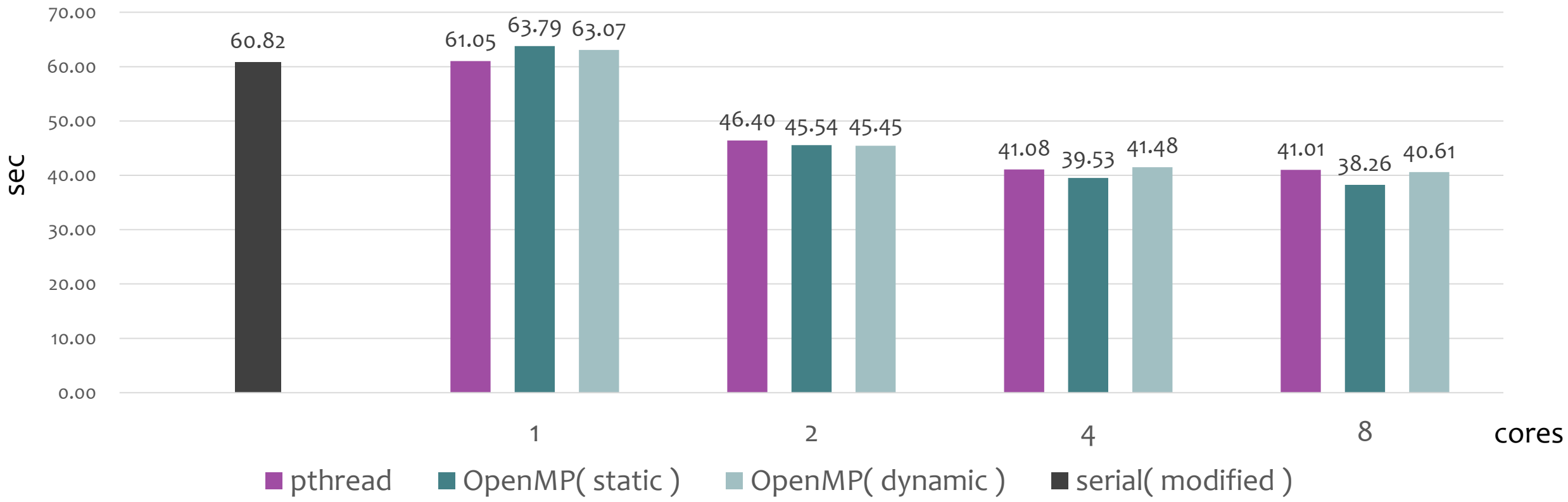
Environment setting

- CPU: Ubuntu 16.04, i7-4790k(4C8T 4.0GHz), 16GB
- Compiler optimization flag: -O3
- GPU: GEFORCE GTX 1060 6GB
- APIs: OpenCV 3.3.1
- CUDA: 8.0
- Input: 1920x1080 1min (30 fps)

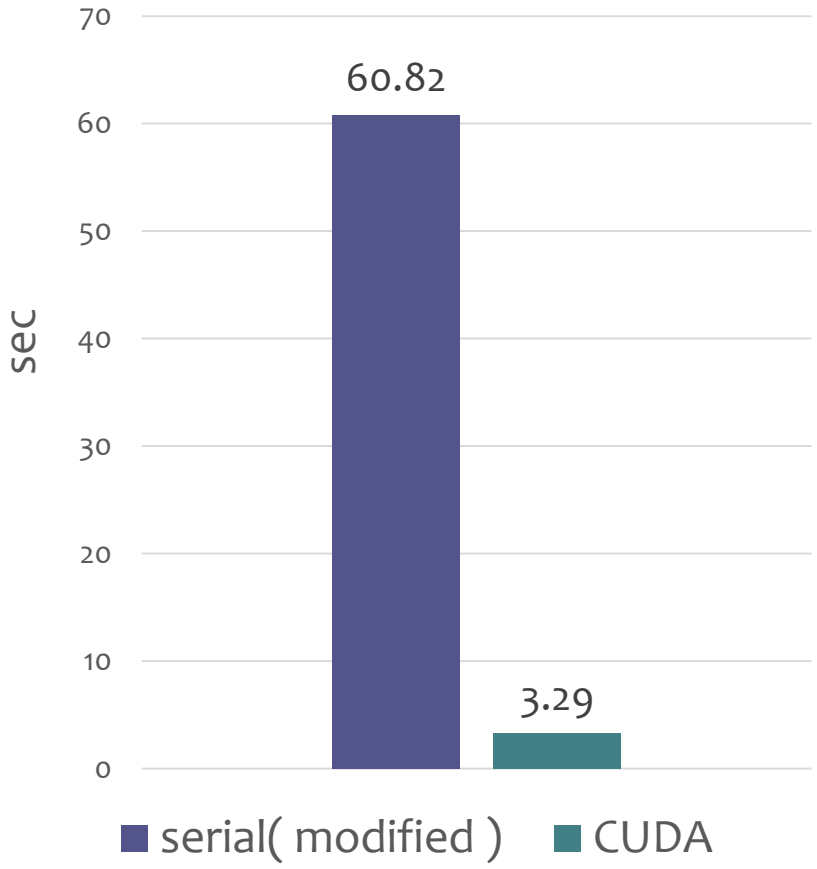
Proposed solution

- Pthreads
- OpenMP
- CUDA
- OpenCL

Evaluation- multi-thread

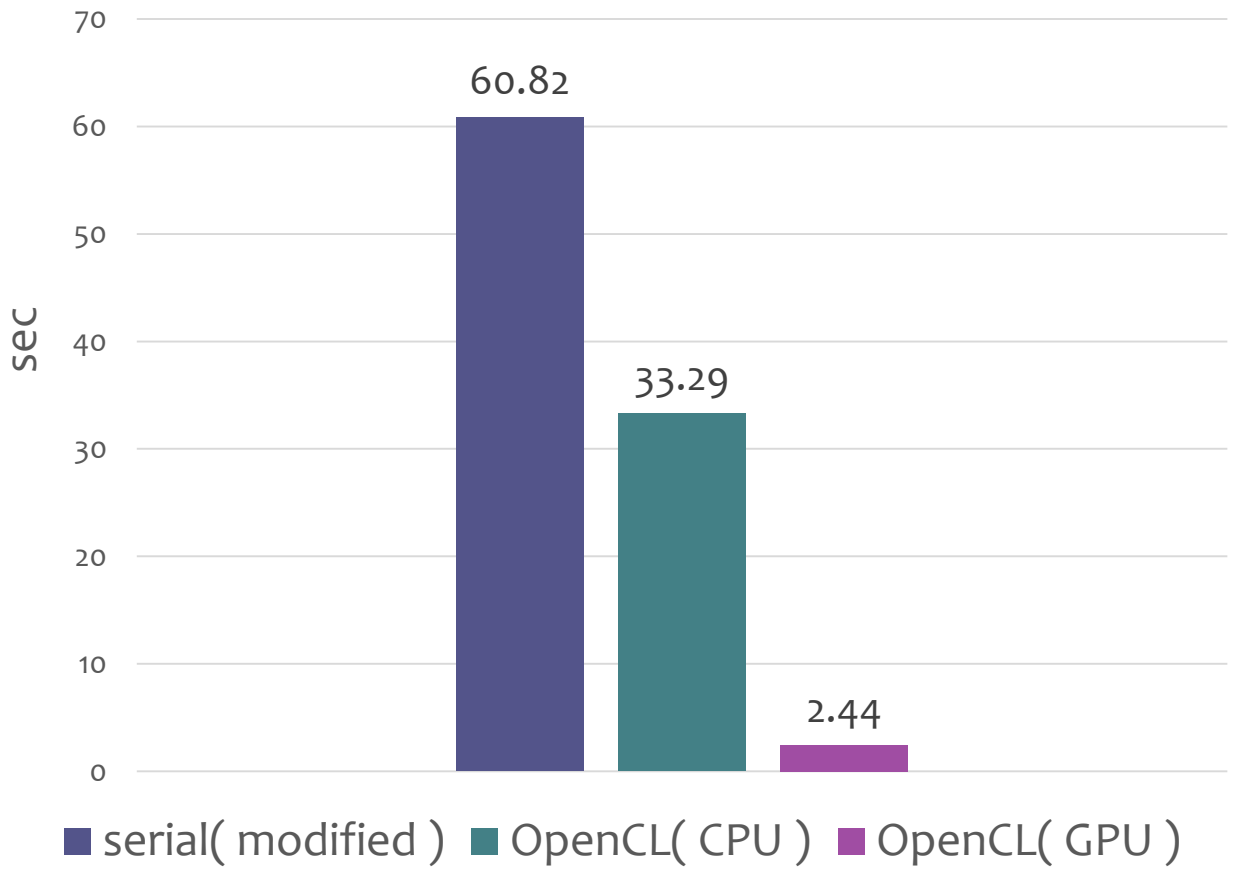


Evaluation- CUDA



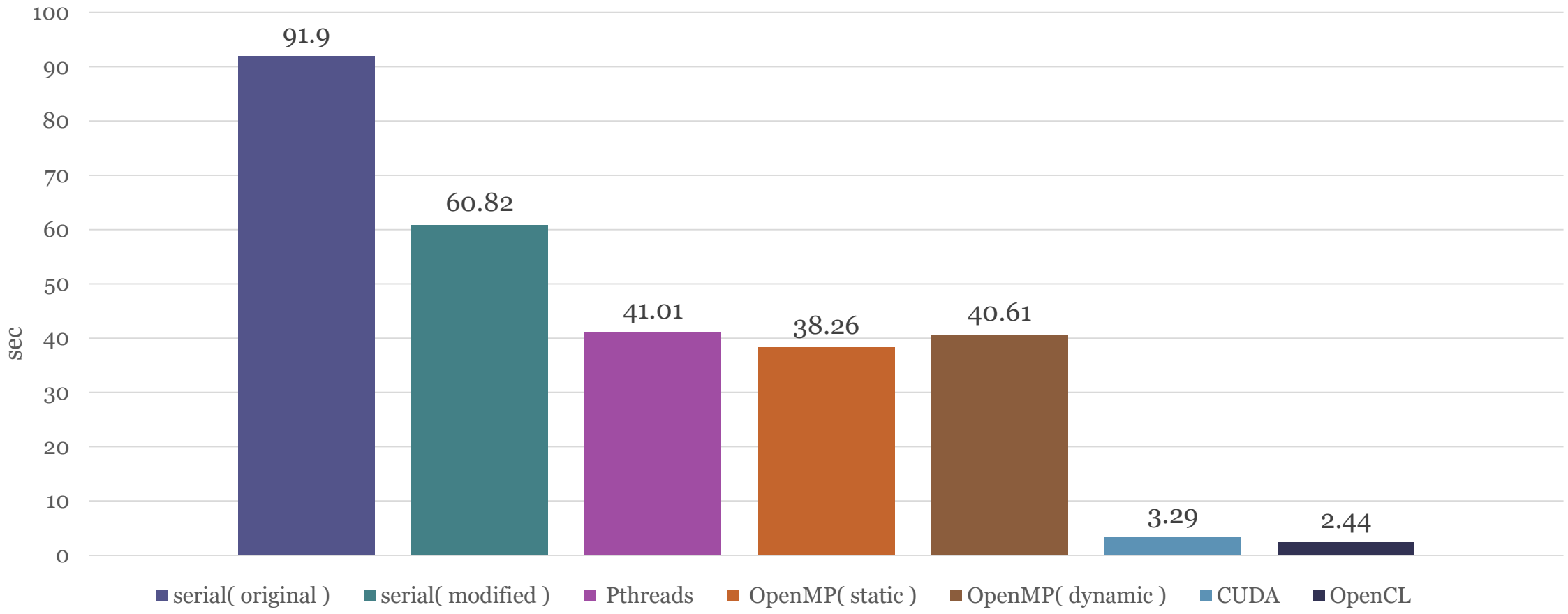
~18.5x Speedup

Evaluation- OpenCL



~ 24.9x Speedup

Final result



Conclusion

- The parallelism of GPU is powerful.
- In our project, OpenCL performances better than CUDA.
  - In OpenCL implementation, OpenCV built-in API for OpenCL is used
  - We consider these API perform some optimization not in our CUDA implementation, including memory copy etc.

Contribution

- 0556055 吳雅就 : OpenCL
- 0656017 顏義洋 : CUDA
- 0656132 巫謹佑 : Pthread, OpenMP

Q & A