

# Lập trình song song trên GPU

## BT1: CUDA cơ bản



---

Nên nhớ mục tiêu chính ở đây là **học, học một cách chân thật**. Bạn có thể thảo luận ý tưởng với bạn khác, nhưng **bài làm phải là của bạn, dựa trên sự hiểu thật sự của bạn**. **Nếu vi phạm thì sẽ bị 0 điểm cho toàn bộ môn học**.

---

### Đề bài

Viết chương trình làm mờ ảnh RGB (đã hướng dẫn trên lớp).

Mình có đính kèm:


- File ảnh đầu vào “in.pnm” (trong Windows, bạn có thể xem file \*.pnm bằng chương trình [IrfanView](#)). File này đã được nén lại thành “in.zip”. Bạn nên copy file nén này qua server rồi mới giải nén bằng câu lệnh `unzip in.zip`; nếu bạn giải nén ra file pnm rồi mới copy qua thì sẽ rất nặng, và thường sẽ bị đứt kết nối giữa chừng. (Với file ảnh kết quả cũng vậy, chẳng hạn nếu bạn muốn mở xem file “out\_host.pnm” thì đầu tiên bạn nên nén lại ở server bằng câu lệnh `zip out_host.zip out_host.pnm` rồi mới copy về và mở xem.)
- File khung chương trình “bt1.cu”. Chương trình sẽ:
  - Đọc file ảnh đầu vào RGB.
  - Làm mờ ảnh đầu vào bằng host (làm tuần tự).
  - Làm mờ ảnh đầu vào bằng device (làm song song). *Bạn sẽ phải code phần này, cụ thể là ở những chỗ mình để “// TODO”. Lưu ý: bạn cần kiểm lỗi khi gọi các hàm CUDA API (dùng macro `CHECK` mà mình đã viết sẵn cho bạn) và khi gọi hàm kernel.*
  - So sánh ảnh kết quả của device với host để giúp bạn biết là đã cài đặt đúng hay chưa.
  - Ghi các ảnh kết quả xuống file.

### Hướng dẫn về các câu lệnh

- Biên dịch file “bt1.cu”: `nvcc bt1.cu -o bt1`  
Câu lệnh này sẽ biên dịch file “bt1.cu” bằng trình biên dịch nvcc của NVIDIA, và xuất ra file chạy “bt1” (nếu bạn muốn xuất ra file chạy có tên khác thì sau `-o` bạn thay `bt1` bằng tên mà bạn muốn; nếu bạn chỉ gõ là `nvcc bt1.cu` thì sẽ xuất ra file chạy có tên mặc định là “a.out”)
- Chạy file “bt1” với file ảnh đầu vào là “in.pnm” và xuất ảnh kết quả ra file “out.pnm” (kết quả của host sẽ xuất ra file “out\_host.pnm”, còn device thì là “out\_device.pnm”): `./bt1 in.pnm out.pnm`  
Chương trình sẽ in ra thời gian thực thi của host và của device, và giá trị khác biệt trung bình giữa ảnh kết quả của host và của device (được tính bằng cách: lấy các pixel tương ứng của 2 ảnh trừ cho nhau, lấy trị tuyệt đối, và cuối cùng tính trung bình hết lại). Như vậy, nếu giá trị khác biệt trung bình bằng 0 thì nghĩa là 2 ảnh giống hệt nhau. Nếu giá trị khác biệt trung bình có giá trị nhỏ (ví dụ, 0.xxx) thì chưa chắc là sai, vì khi tính toán số thực thì giữa CPU và GPU có thể có sai biệt nhỏ; nhưng nếu giá trị khác biệt trung bình lớn hơn 0.xxx thì chắc là sai rồi.

Mặc định thì sẽ dùng block có kích thước 32×32; nếu bạn muốn chỉ định kích thước block thì truyền thêm vào câu lệnh 2 con số lần lượt ứng với kích thước theo chiều x và theo chiều y của block (ví dụ, `./bt1 in.pnm out.pnm 32 16`).

### File báo cáo

- Ghi họ tên và MSSV ở đầu file.
- Chụp lại màn hình kết quả chạy với các kích thước block khác nhau: 8x8, 16x16, 32x32, 64x64 (nếu kiểm lỗi đúng thì sẽ bắt được lỗi khi chạy với block 64x64). Để chụp lại một phần màn hình, trong Windows 10, bạn có thể ấn +Shift+S, rồi ấn giữ chuột trái và kéo chọn vùng cần chụp, rồi Ctrl+V vào file word là xong.

### Nộp bài

Trong thư mục <MSSV> (vd, nếu bạn có MSSV là 1234567 thì bạn đặt tên thư mục là 1234567), bạn để:

- File code “bt1.cu”
- File báo cáo “bt1.pdf”

Sau đó, bạn nén thư mục này lại và nộp ở link trên moodle.