Modello di Programmazione CUDA
Esercitazione

Sistemi di Elaborazione Accelerata, Modulo 2

A.A. 2025/2026

Fabio Tosi, Università di Bologna

Obiettivi della sessione

- ✓ Consolidare scrittura kernel e calcolo indici
- ✓ Sperimentare configurazioni griglia/blocchi
- ✓ Familiarizzare con profiling (Nsight Systems/Compute)

* Starter Kit (.zip) nella cartella “Codice CUDA” in Virtuale

Esercizio 1: Rotazioni 90°

Problema:

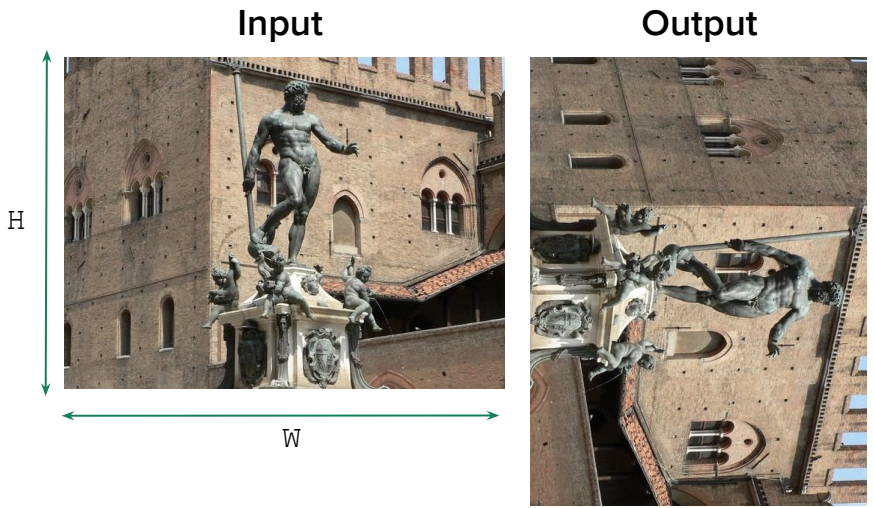
- Data un'immagine H×W, implementare **rotazioni di 90°** sia in **senso orario** che **antiorario**.

📝 Richieste

- Determinare il **mapping** coordinate input ➜ output
- **Scrivere il kernel** per entrambe le trasformazioni.
- Implementare versione **CPU** per confronto.
- Verificare **correttezza** confrontando GPU vs CPU.

🔬 Eseperimenti con Nsight

- Profilare il kernel per **diverse configurazioni di blocchi** (4×4, 16×16, 32×32).
- Misurare **tempo kernel** e **trasferimenti** per immagini di differente risoluzione.
- **Annotare alcune metriche** (es. Achieved Occupancy, DRAM Throughput). Le metriche cambiano con la dimensione dell'i**mmagine?**
- Quale configurazione è **più veloce**?



(senso orario)

Esercizio 2: Convoluzione Separabile (Filtro di Media)

Problema:

- Applicare un **filtro di media** (box filter) su un'immagine sfruttando la **separabilità.**

💡 Idea chiave convoluzione separabile

- Un filtro di media N×N ha tutti gli elementi uguali a $1/N^2$ ed è separabile.
- Invece di fare N×N operazioni per pixel (convoluzione 2D), possiamo:
  1. **Passata orizzontale** ➜ Convoluzione 1D lungo ogni riga con un filtro 1 x N
  2. **Passata verticale** ➜ Convoluzione 1D lungo ogni colonna con un filtro N x 1 sul risultato intermedio.
- Risultato finale identico, ma solo 2N operazioni invece di N^2! Ottimizzazione algoritmica: da $O(N^2)$ a $O(2N)$

📝 Richieste

- Sperimentare la **convoluzione 2D naïve** (N×N, già vista a lezione)
- **Scrivere kernel per passata orizzontale** (convoluzione 1D su righe)
- **Scrivere kernel per passata verticale** (convoluzione 1D su colonne)
- **Verificare** che i risultati coincidano (separabile = 2D naïve)

🔬 Eseperimenti con Nsight

- Testare filtri di media 3×3, 5×5, 7×7 e 9x9.
- Confrontare tempi: separabile vs 2D naïve.
- Provare su immagini di dimensione differente

Esercizio 2: Convoluzione Separabile (Filtro di Media)

Esempio pratico passo passo (3×3)

Kernel 2D 3x3

$$K_{2D} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Il filtro può essere scritto come prodotto esterno di due filtri 1D: uno **orizzontale** e uno **verticale**:

$$K_{2D}(i,j) = K_h(i) \cdot K_v(j)$$

Kernel Separabili

$$K_h = \frac{1}{3}[1,1,1], \quad K_v = \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

1. Prima si applica il **kernel orizzontale su ogni riga** ➜ matrice intermedia
2. Poi si applica il **kernel verticale su ogni colonna** della matrice intermedia ➜ output finale

Input (5x5)

$$\mathbf{I} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{bmatrix}$$

Passata orizzontale

$$\mathbf{I}_h = \begin{bmatrix} 1.0 & 2.0 & 3.0 & 4.0 & 3.0 \\ 7.0 & 8.0 & 9.0 & 10.0 & 9.0 \\ 12.0 & 13.0 & 14.0 & 15.0 & 14.0 \\ 17.0 & 18.0 & 19.0 & 20.0 & 19.0 \\ 22.0 & 23.0 & 24.0 & 25.0 & 24.0 \end{bmatrix}$$

Passata verticale

$$\mathbf{I}_{out} = \begin{bmatrix} 3.0 & 4.0 & 5.0 & 6.0 & 5.0 \\ 8.0 & 9.0 & 10.0 & 11.0 & 10.0 \\ 13.0 & 14.0 & 15.0 & 16.0 & 15.0 \\ 18.0 & 19.0 & 20.0 & 21.0 & 20.0 \\ 17.0 & 18.0 & 19.0 & 20.0 & 19.0 \end{bmatrix}$$

- Risultato identico alla convoluzione 2D completa, ma con **meno operazioni per pixel**.

Esercizio 3: Grayscale su Video

Problema:

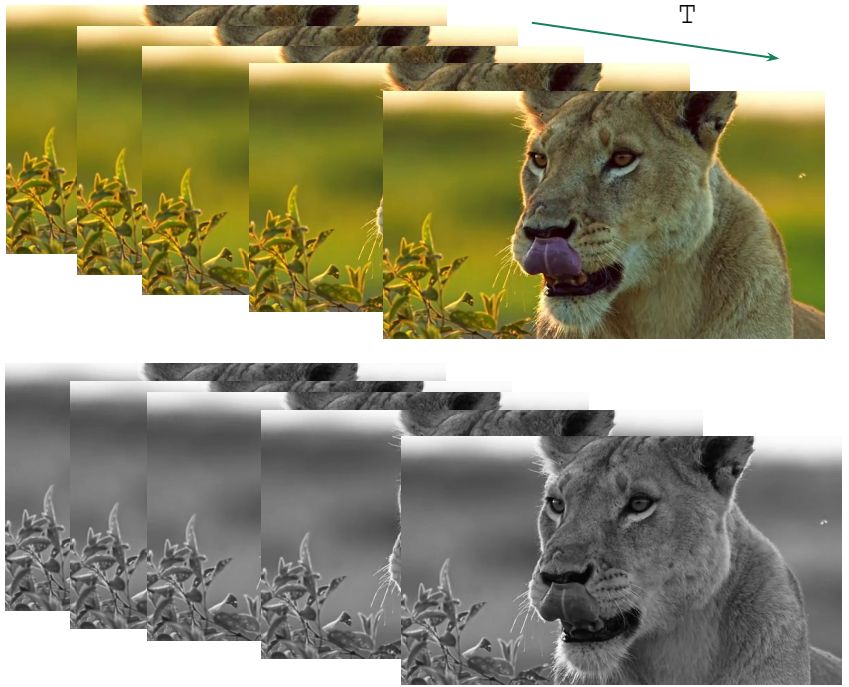
- **Convertire un video RGB** con T frame di dimensione H×W in grayscale.
- Trattare il video come **array 3D** [T][H][W][3]: ogni frame è un'immagine da convertire.

📝 Richieste

- Scrivere il kernel CUDA per la conversione.
- Gestire i frame con loop (2D grid) o griglia 3D (dim3(W,H,T)).
- Verificare **correttezza** confrontando GPU vs CPU

🔬 Eseperimenti con Nsight

- Confrontare tempi 2D+loop vs 3D.
- Misurare **tempo kernel** e **trasferimenti** per video di differente risoluzione.
- **Annotare alcune metriche** (es. Achieved Occupancy, DRAM Throughput). Le metriche cambiano con la dimensione del video**?**
- **Come scala il tempo** con il numero di frame?



Esercizio 4: Downsampling (Resize)

Problema:

- **Ridimensionare un'immagine** H×W a (H/2)×(W/2) facendo la media dei pixel in ogni area 2×2 dell'immagine originale.

📝 Richieste

- Determinare il mapping coordinate input ➜ output
- **Scrivere il kernel** CUDA per la riduzione 2×2 ➜ 1.
- Implementare versione **CPU naïve** per confronto
- Verificare **correttezza** confrontando GPU vs CPU

🔬 Eseperimenti con Nsight

- Profilare il kernel per diverse configurazioni di blocchi (4×4, 16×16, 32×32).
- Misurare **tempo kernel** e **trasferimenti** per immagini di differente risoluzione.
- **Annotare alcune metriche** (es. Achieved Occupancy, DRAM Throughput). Le metriche cambiano con la dimensione dell'i**mmagine?**
- Ripetere con **downsampling per salto** (primo pixel di ogni blocco anziché media) e confrontare i tempi.