Programmazione Funzionale e Parallela (A.A. 2016-2017)

Corso di Laurea in Ingegneria Informatica e Automatica Sapienza Università di Roma



Esame del 23/01/2017 – Durata 1h 30' (solo esonerati)

Inserire nome, cognome e matricola nel file studente.txt.

Esercizio 1 (Filtri grafici mediante OpenCL)

Lo scopo dell'esercizio è quello di scrivere un modulo C basato su OpenCL che, data in input un'immagine a 256 toni di grigio di dimensione $w \times h$, crei una nuova immagine della stessa dimensione, come nell'esempio sotto.



(a) Immagine originale



(b) Immagine dopo creazione art poster

Si completi nel file art/art.c la funzione art con il seguente prototipo:

dove:

- in: puntatore a un buffer di dimensione w*h*sizeof(unsigned char) byte che contiene l'immagine di input in formato row-major¹;
- w: larghezza di in in pixel (numero di colonne della matrice di pixel);
- h: altezza di in in pixel (numero di righe della matrice di pixel);
- out: puntatore a puntatore a buffer di dimensione 2w*2h*sizeof(unsigned char) byte che deve contenere l'immagine di output in formato row-major; il buffer deve essere allocato nella funzione art.

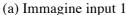
Per compilare usare il comando make. Per effettuare un test usare make test. Verrà prodotta l'immagine di output colosseo-art.pgm.

¹ Cioè con le righe disposte consecutivamente in memoria.

Esercizio 2 (Elaborazione di immagini mediante vettorizzazione SSE)

Lo scopo dell'esercizio è quello di scrivere un modulo C basato su vettorizzazione SSE che, date in input due immagini a 256 toni di grigio di dimensione rispettivamente $w1 \times h1$ e $w2 \times h2$ crei una nuova immagine ottenuta fondendole come nell'esempio seguente:







(b) Immagine input 2



(c) Immagine output

L'immagine di output ha dimensioni pari al minimo delle righe/colonne delle due immagini di input. La fusione avviene alternando pixel dalle due immagini, partendo dalla prima sulle righe di indice pari e dalla seconda sulle righe di indice dispari.

Si completi nel file blend/blend.c la funzione blend con il seguente prototipo:

dove:

- in1: puntatore a un buffer di dimensione w1*h1*sizeof(unsigned char) byte che contiene la prima immagine di input in formato row-major;
- in2: puntatore a un buffer di dimensione w2*h2*sizeof(unsigned char) byte che contiene la seconda immagine di input in formato row-major;
- out: puntatore a buffer di dimensione (*w)*(*h)*sizeof(unsigned char) byte che deve contenere l'immagine di output in formato row-major; il buffer deve essere allocato nella funzione blend;
- w: puntatore a buffer dove scrivere la larghezza in pixel dell'immagine di output;
- h: puntatore a buffer dove scrivere l'altezza in pixel dell'immagine di output.

Per compilare usare il comando make. Per effettuare un test usare make test. Verrà prodotta l'immagine di output blend.pgm.

Ai fini della soluzione si usi la funzione SSE4.1:

```
__m128i _mm_blendv_epi8(__m128i a, __m128i b, __m128i mask)
```

che restituisce un packed integer res tale che (immaginando res, a e b come array di 16 char) res[i]=a[i] se mask[i]<128 e res[i]=b[i] altrimenti, con i=0..15.