Ingegneria degli Algoritmi (A.A. 2013-2014)

Corsi di Laurea in Ingegneria Informatica e Automatica, Ingegneria dei Sistemi Informatici, e Laurea Magistrale in Ingegneria Informatica

Sapienza Università di Roma

Esercitazione di laboratorio (10/06/2014) – Durata 2h 30'

Preliminari (aula 17)

1. Fare login in Linux (Scientific Linux 5.1) con le credenziali: `studente` (login) e `informatica` (password)
2. Creare sulla scrivania (`/local/studente/Desktop`) una directory chiamata: `ida2014`
3. Entrare nella directory `/local/studente/Desktop/ida2014`
4. Copiare il testo e il codice fornito per l'esercitazione nella directory di lavoro: `cp -rf /home/distrib/demetres-sw/eserc140610 .`
5. La directory `/local/studente/Desktop/ida2014/eserc140610` contiene:
    a. `addframe`: directory di lavoro per l'esercitazione
    b. `esempi`: esempi di programmi OpenCL da cui trarre ispirazione
    c. `docs`: documentazione utile su OpenCL

Esercizio 1

Lo scopo dell'esercitazione è quella di scrivere un modulo C che, data in input una immagine a toni 256 di grigio di dimensione $w \times h$, un parametro `frame_size` e un tono di grigio `tone`, crei una nuova immagine allocata dinamicamente ottenuta da quella di input aggiungendo una cornice di spessore `frame_size` e punto di grigio `tone` $\in [0, 255]$, come nell'esempio sotto.

(a) Immagine originale a 256 toni di grigio di dimensione 500×334

(b) Immagine con cornice di spessore `frame_size`=20 pixel e `tone`=40

Si vada nella directory di lavoro `addframe` e si definisca nel file `addframe.c` la funzione `addframe` con il seguente prototipo:

```
void addframe(unsigned char* in, int w, int h,
              int frame_size, unsigned char tone,
              unsigned char** out, int* ow, int* oh,
              clut_device* dev, double* td);
```

dove:

- `in`: puntatore a un buffer di dimensione `w*h*sizeof(unsigned char)` byte nella memoria dell'host che contiene l'immagine di input in formato row-major;
- `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);
- `frame_size`: spessore in pixel della cornice;
- `tone`: punto di grigio della cornice (0=nero, 255=bianco);
- `out`: parametro in cui restituire il puntatore all'immagine di output in formato row-major, che la funzione deve allocare dinamicamente;
- `ow`: parametro in cui restituire la larghezza dell'immagine di output in pixel;
- `oh`: parametro in cui restituire l'altezza dell'immagine di output in pixel;
- `dev`: ambiente di esecuzione della GPU (si veda `clut.h`);
- `td`: parametro in cui restituire la durata dell'esecuzione del kernel.

Scrivere un opportuno kernel OpenCL nel file `addframe.cl`.

Suggerimento: eseguire il kernel su un NDRange a due dimensioni dove la dimensione 0 corrisponde all'asse orizzontale (x) e la dimensione 1 corrisponde all'asse verticale (y).

Compilazione e test.

Directory di lavoro: `~/Desktop/ida2014/eserc140610/addframe/`

1. Compilazione programma di test (*una tantum*): dare il comando `make`, che genera il file eseguibile `addframe`
2. Compilazione ed esecuzione kernel:
   a. `make test1`: primo test su immagine 500×334 a 256 toni di grigio del Colosseo;
   b. `make test2`: secondo test su immagine 500×334 a 256 toni di grigio del Colosseo;
   c. `make test3`: test su una versione 512×512 a 256 toni di grigio del classico bechmark "Lena" usato in computer graphics;

Nella directory `results` verranno generate varie immagini ottenute aggiungendo diverse cornici alle immagini date in input. Le immagini, salvate in formato PGM (Portable Graymap Format), possono essere visualizzate con il programma Gimp.