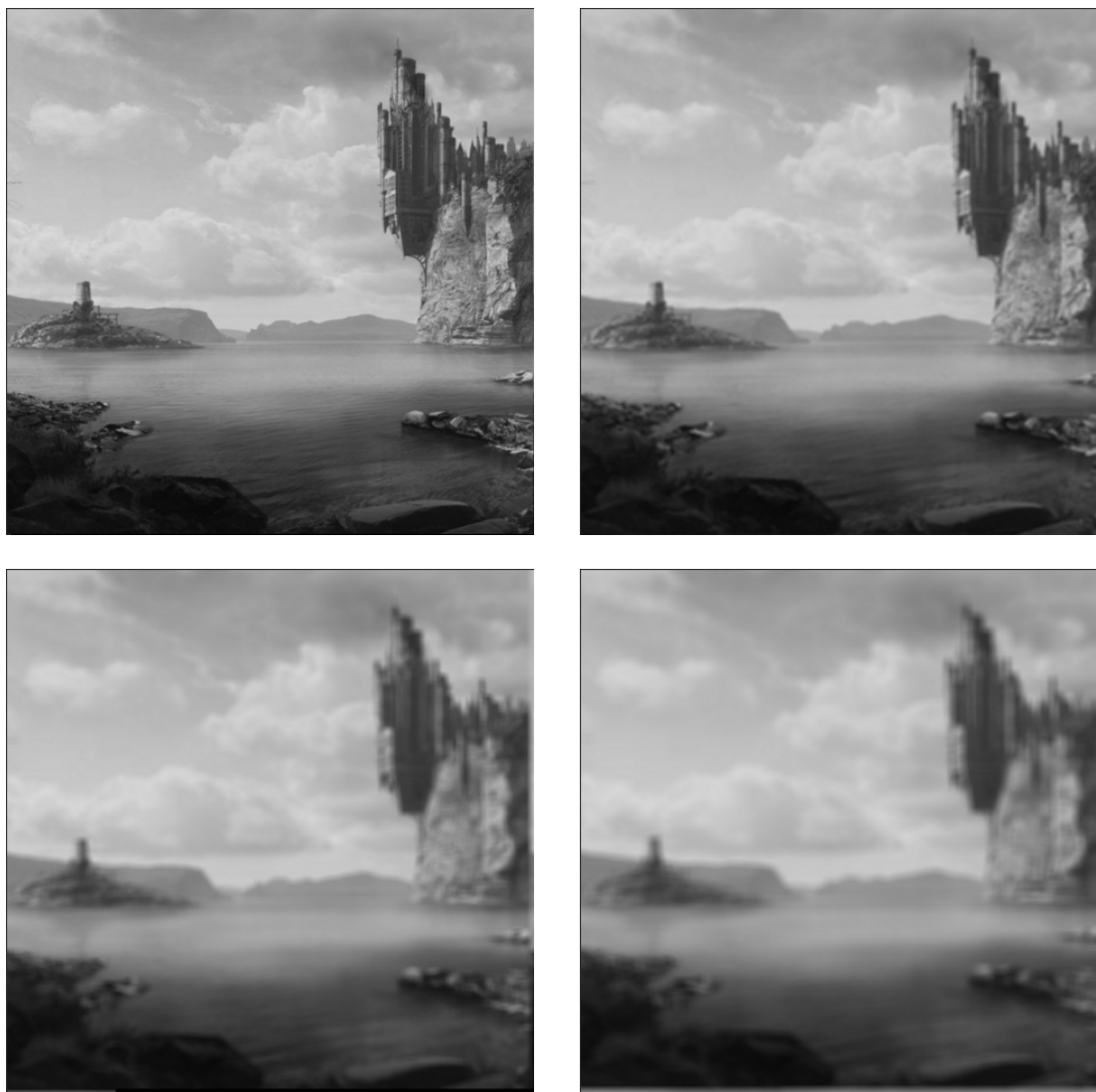


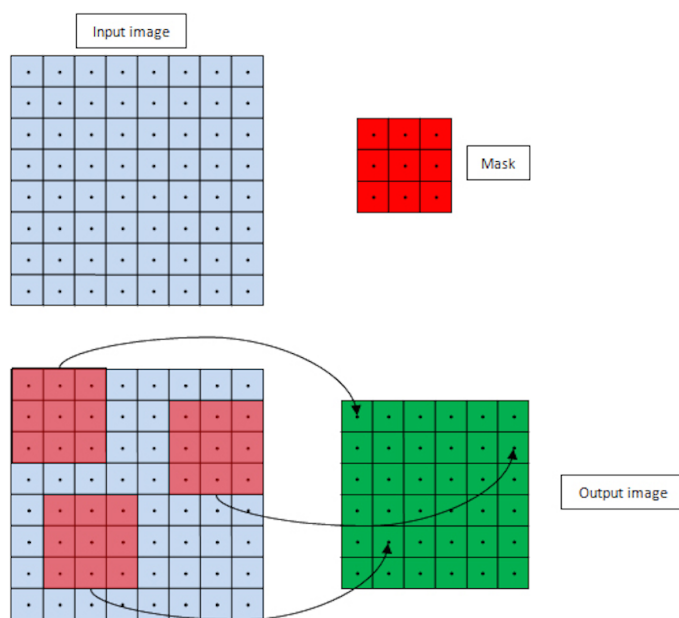
HOMEWORK 2: IMAGE BLURRING

In questo homework si richiede di scrivere codice OpenCL per applicare a un'immagine a toni di grigio un filtro grafico di blurring, effettuando il calcolo su GPU. Il blurring permette di sfocare un'immagine come mostrato nella seguente figura, in cui all'immagine originale (in alto a sinistra) sono applicati tre differenti filtri di blurring che permettono di ottenere un effetto di sfocatura via via maggiore:



Realizzazione del blurring. Supponete che l'immagine originaria sia memorizzata in una matrice di dimensioni $h \times w$, ovvero con h righe e w colonne. Il blurring si ottiene applicando alla matrice in input un filtro, rappresentato da una matrice quadrata $s \times s$, con s dispari. Nell'esempio mostrato sopra, le tre immagini sfocate sono ottenute applicando alla matrice originaria filtri di lato $s = 3, 15$ e 19 , rispettivamente. Maggiore è il valore di s , maggiore è la sfocatura ottenuta. In letteratura, il filtro è chiamato anche *mask* o *kernel*, quest'ultimo da non confondere con le funzioni kernel in OpenCL.

Il filtro è applicato su un pixel (i, j) dell'immagine originaria sovrapponendo concettualmente a (i, j) il centro della matrice filtro (che esiste certamente essendo s dispari), come schematizzato di seguito:



Dimensioni dell'immagine di output. Si osservi che la matrice di output è più piccola rispetto a quella di input. Infatti, nella nostra versione semplificata del problema, i pixel sul bordo (prima riga, ultima riga, prima colonna e ultima colonna) non sono stati considerati in quanto non è possibile sovrapporre il centro della matrice filtro ai pixel contenuti in queste posizioni rimanendo all'interno dell'immagine. Si osservi anche che più grande è il filtro, più piccola è l'immagine di output ottenuta: ad esempio, se $s = 5$, sarà necessario rimuovere le due righe più in alto, le due righe più in basso, le due colonne più a sinistra e le due colonne più a destra, ottenendo un'immagine di output di dimensioni 4×4 .

Valori dei pixel di output. Il valore di un pixel di output viene determinato combinando opportunamente i valori delle due matrici di dimensioni $s \times s$ (la sottomatrice dell'immagine di input e il filtro) tramite una somma di prodotti: i valori dei pixel in posizioni corrispondenti nelle due sottomatrici vengono moltiplicati, e gli $s \times s$ prodotti vengono sommati tra loro. Un esempio è illustrato di seguito:



Il pixel della matrice di output corrispondente a quello evidenziato con il rettangolo rosso è ottenuto come $(164 \times 0 + 188 \times 1 + 164 \times 0 + 178 \times 1 + 201 \times 1 + 197 \times 1 + 174 \times 0 + 168 \times 1 + 181 \times 0)/5 = 932/5$. Si noti che la somma

di prodotti è *normalizzata* per il numero di 1 presenti nella matrice filtro, pari a 5 nell'esempio.

Attenzione: il prodotto effettuato non ha nulla a che vedere con il prodotto di matrici. Diversamente dal prodotto scalare riga×colonna usato nella moltiplicazione di matrici, nel caso del blurring i valori moltiplicati sono quelli di pixel in posizioni corrispondenti: il valore in posizione (x, y) nel filtro è moltiplicato con il pixel in posizione (x, y) nella sottomatrice dell'immagine di input.

Note implementative. Un'implementazione che utilizzi solo memoria globale è piuttosto semplice da ottenere. Tuttavia l'uso di memoria locale – sebbene un po' più complesso – potrebbe portare grandi benefici in termini di prestazioni, soprattutto all'aumentare delle dimensioni del filtro. Sebbene non strettamente necessario ai fini dello svolgimento dell'homework, l'eventuale uso di memoria locale sarà valutato molto positivamente.

Lettura dell'input e scrittura dell'output. L'applicazione OpenCL realizzata deve caricare da file un'immagine di input a toni di grigio in formato `pgm` e salvare su file l'immagine di output ottenuta effettuando il blurring. Ciascun pixel è un valore tra 0 e 255 e può essere rappresentato con un `unsigned char`. Il codice per leggere e scrivere un'immagine è dato nei file `pgm.h` e `pgm.c`. Alcune immagini su cui eseguire test sono messe a disposizione sul sito del corso.

Esperimenti. A corredo dell'implementazione, eseguite una serie di esperimenti e misurate il tempo di esecuzione richiesto da ciascuno di essi. Confrontate i tempi richiesti con quelli di una esecuzione sequenziale, valutando empiricamente lo speedup ottenuto. Studiate inoltre come variano le prestazioni all'aumentare delle dimensioni dei filtri usando i seguenti filtri (per ciascuno di essi, ricordate di normalizzare opportunamente i pixel di output come descritto in precedenza):

$$\begin{array}{ccc}
 3 \times 3 & 5 \times 5 & 7 \times 7
 \end{array}
 \qquad
 \begin{array}{ccc}
 \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}
 \end{array}$$

$$\begin{array}{ccc}
 9 \times 9 & \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}
 \end{array}$$

Documentazione. A corredo dell'implementazione (un archivio compresso contenente tutti i file necessari per la compilazione), inviatemi una breve relazione in formato pdf che illustri:

- le caratteristiche salienti dell'implementazione realizzata e le relative ottimizzazioni;
- le caratteristiche della/e piattaforma/e su cui avete eseguito l'analisi sperimentale;
- i risultati sperimentali ottenuti, tramite tabelle e grafici opportunamente commentati;
- alcuni esempi di immagini cui siano stati applicati i filtri di blurring;
- la modalità per compilare ed eseguire il programma (da linea di comando).

Modalità di svolgimento: potete lavorare in gruppi composti da al più 3 persone.

Deadline: data in cui si svolge la prova scritta nell'appello in cui volete sostenere/verbalizzare l'esame.