SISTEME TOLERANTE LA DEFECTE
Tema #1 Filtrarea imaginilor

Termen de predare: 04-Apr-2021 23:55

Responsabil Temă: **Bureacă Emil**

Obiective

Scopul acestei teme este de a implementa, în C, folosind biblioteca MPI, un sistem distribuit având ca scop filtrarea de imagini.

Date introductive

Obiectivul temei este de a crea un program MPI scalabil ce oferă funcționalitatea de aplicare a unor filtre asupra imaginilor. Filtrele reprezintă o componentă de bază în domeniul procesării imaginilor și ajută la sporirea anumitor caracteristici ale acestora.

În cazul acestei teme, vom aborda filtrarea liniară. Această operație constă în suprapunerea unei măști de filtrare peste fiecare pixel al imaginii originale, astfel încât centrul măștii să corespundă cu pixelul curent. Folosind masca se va calcula o nouă valoarea a pixelului. Noua valoare este egală cu suma tuturor produselor între coeficienții măștii și valorile pixelilor peste care se suprapun.

Pentru a simplifica lucrurile, vom considera o mască de filtrare de formă pătrată, cu dimensiunea de 3x3.

$$\mathrm{M} = \begin{bmatrix} m_{-1,-1} & m_{-1,0} & m_{-1,1} \\ m_{0,-1} & \mathbf{m}_{0,0} & m_{0,1} \\ m_{1,-1} & m_{1,0} & m_{1,1} \end{bmatrix}$$

Procesul de filtrare

Programul va primi ca input o imagine și va aplica unul sau mai multe filtre: **smooth**, **blur**, **sharpen**, **mean**, **emboss**.

Deoarece filtrarea se aplică asupra pixelului și a vecinilor săi, apare cazul special al marginilor. Pixelii marginali nu au toți cei opt vecini. În practică, se adaugă un chenar imaginii format din pixeli de o anumită culoare, de exemplu alb sau negru în așa fel încât filtrarea să poată fi aplicată pe întreaga imagine. Totuși, pentru această temă, valoarile pixelilor marginilor vor rămâne nemodificate.

În continuare, vom defini, prin intermediul măștilor de filtrare, următoarele filtre:

1. **Filtrul smooth**

$$\mathrm{M} = \frac{1}{9} * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

2. **Filtrul Gaussian blur**

$$\mathrm{M} = \frac{1}{16} * \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

3. **Filtrul sharpen**

$$M = \frac{1}{3} * \begin{bmatrix} 0 & -2 & 0 \\ -2 & 11 & -2 \\ 0 & -2 & 0 \end{bmatrix}$$

4. **Filtrul mean**

$$M = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

5. **Filtrul emboss**

$$M = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

Observații

- Programul poate primi imagini în format **.pnm** sau **.pgm**, iar formatul de fișierului de ieșire este același cu cel de intrare. Pentru a converti imaginile de la un format la altul (e.g. .pgm to .jpg), puteti accesa `https://convertio.co/`. Alternativ puteți folosi linia de comanda folosind comanda **convert** din suita **imagemagick**.
- Deoarece comunicația este un factor de importanță majoră în cadrul sistemelor distribuite, această trebuie realizată într-un mod optim. Cât mai puține puține date transmise grupate în cât mai puține transmiteri.
- Aplicația va trebui să aplice cel puțin un filtru.
- Pot apărea diferențe la calcule. Folosiți tipul **float** pentru măștile de filtrare.
- Pixelii marginali vor ramâne constanți.
- Imaginile pot avea formatele **.pnm** sau **.pgm**.
  Formatul va fi de forma:

```
P(5 sau 6)\n
WIDTH HEIGHT\n
MAXVAL (maxim 255)\n
WIDTH * HEIGHT * NUM bytes reprezentand imaginea (*3 pentru color)
```

  Unde, **P5** se referă la faptul că imaginea este în nuanțe de gri, iar **P6** indică o imagine coloră.
- Pentru imaginile în nuanțe de gri, fiecare pixel este reprezentat printr-o valoare din intervalul $[0, \mathbf{MAXVAL}]$
- Pentru imaginile colore, fiecare pixel deține cate un byte pentru fiecare canal RGB. **Atenție**: aplicarea filtrării se va realiza separat pentru fiecare canal.

Rularea programului

Rularea programului se va realiza astfel:

```
mpirun --oversubscribe -np NUM_PROCS ./homework IMG_IN IMG_OUT FILT1 FILT2 ... FILTK
```

Unde:

- **NUM_PROCS** - numărul de procese;
- **IMG_IN** - numele fișierului imaginii de intrare;
- **OUT_FILENAME** - numele fișierului imaginii rezultate în urma aplicării filtrelor;
- **FILT1 FILT2 ... FILTK** - denumirile filtrelor { **smooth**, **blur**, **sharpen**, **mean**, **emboss** }

Exemplu:

```
mpirun --oversubscribe -np 2 ./homework tree.pnm tree_smooth.pnm smooth
```

Trimitere și punctare

Distribuția punctajului este următoarea:

- 30 puncte - Output corect program secvențial;
- 70 puncte - Output corect program distribuit și scalabil.

Arhiva .zip trebuie să conțină fișierele "homework.c". O temă care nu compilează va primi 0 puncte. Tema va fi rulată și testată pentru corectitudine și scalabilitate pe un checker al ATM. Link-ul către acest checker va fi transmis ulterior.

Orice încercare de a abuza checker-ul va duce la un punctaj de 0 pe toate temele.