Manipulation d'images PGM en C

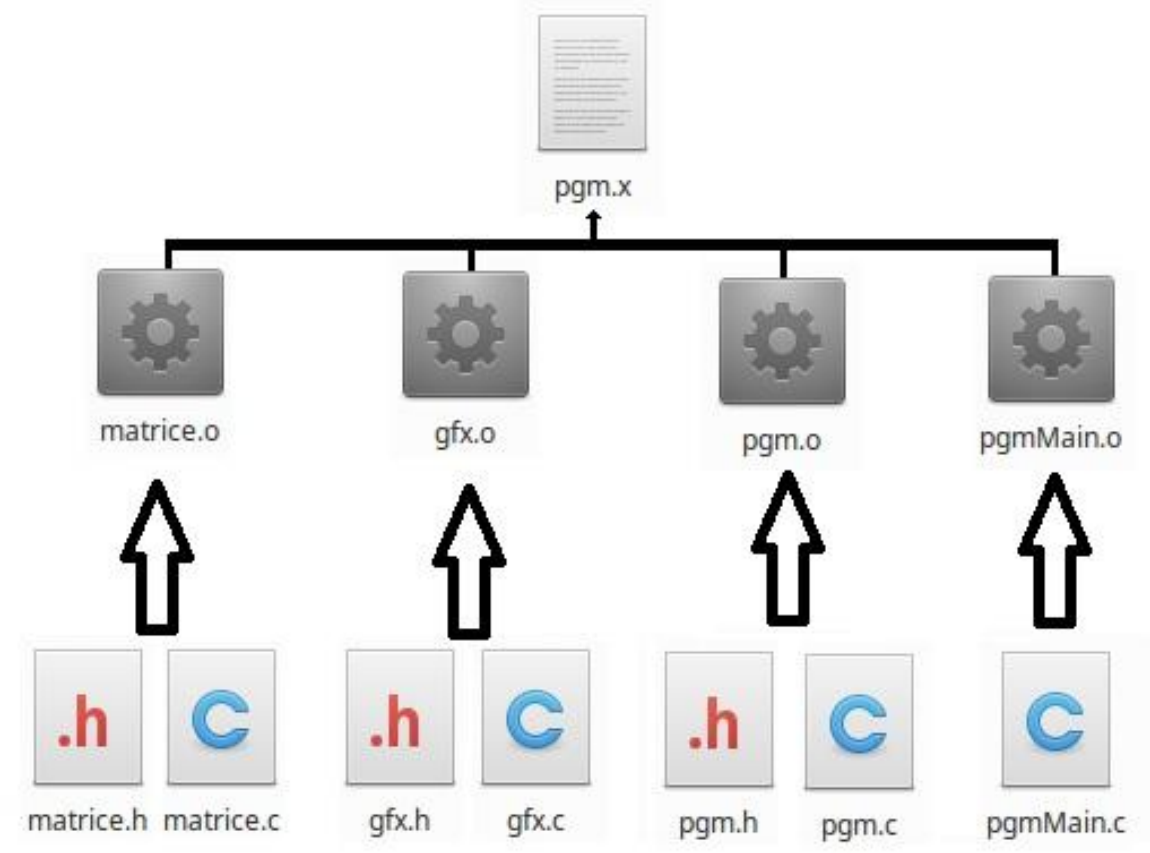
Déroulement

- Présentation générale
- Travail réalisé
- Algo et code
- Démo
- Conclusion
- Questions

Présentation générale

- Ecriture d'un programme en C
- Traitement d'images au format PGM
  - Negatif, crop, symetries, photomaton et filtres
- Affichage d'une image via SDL (exécution)

Présentation générale



Travail réalisé

- Réalisation d’un code en C
- Utilisation d’une librairie sur les matrices
- Utilisation de la librairie SDL2
- Modification d’une image PGM
- Choix parmis plusieurs filtres à appliquer

Algo et code : Le Photomaton

- Rétrécir une image x4
- Pas de perte d'information
  - Uniquement réorganisation de pixels

1,1	1,2	1,3	1,4	1,5	1,6	1,7	1,8
2,1	2,2	2,3	2,4	2,5	2,6	2,7	2,8
3,1	3,2	3,3	3,4	3,5	3,6	3,7	3,8
4,1	4,2	4,3	4,4	4,5	4,6	4,7	4,8
5,1	5,2	**5,3**	**5,4**	5,5	5,6	5,7	5,8
6,1	6,2	**6,3**	**6,4**	6,5	6,6	6,7	6,8
7,1	7,2	7,3	7,4	7,5	7,6	7,7	7,8
8,1	8,2	8,3	8,4	8,5	8,6	8,7	8,8

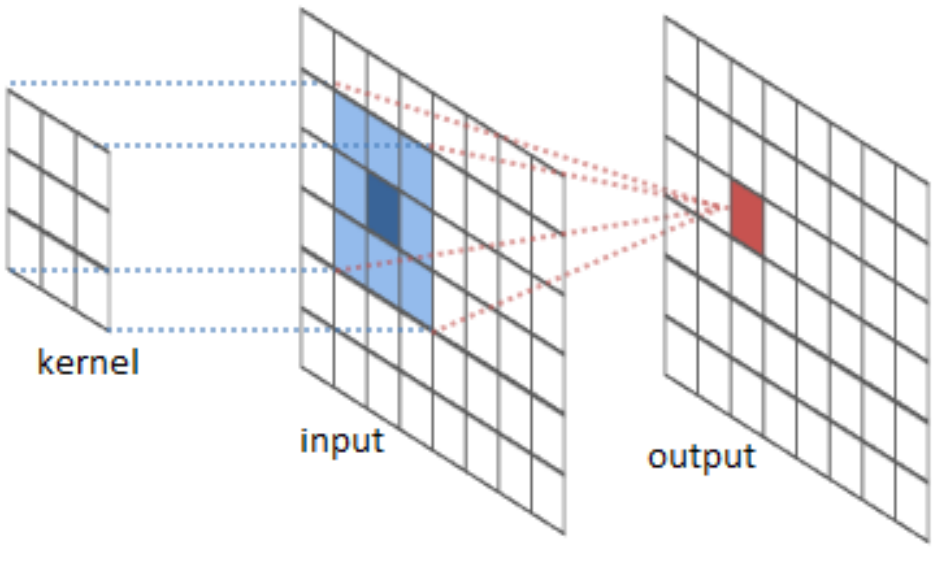
1,1	1,3	1,5	1,7	1,2	1,4	1,6	1,8
3,1	3,3	3,5	3,7	3,2	3,4	3,6	3,8
5,1	**5,3**	5,5	5,7	5,2	**5,4**	5,6	5,8
7,1	7,3	7,5	7,7	7,2	7,4	7,6	7,8
2,1	2,3	2,5	2,7	2,2	2,4	2,6	2,8
4,1	4,3	4,5	4,7	4,2	4,4	4,6	4,8
6,1	**6,3**	6,5	6,7	6,2	**6,4**	6,6	6,8
8,1	8,3	8,5	8,7	8,2	8,4	8,6	8,8

Algo et code : Le Photomaton

```
118  pgm_error pgm_photomaton(pgm *photomaton, const pgm *const orig){
119      if(NULL== orig || NULL==photomaton){
120          return failure;
121      }
122      photomaton->max=orig->max;
123      matrix_alloc(&(photomaton->pixels),orig->pixels.m,orig->pixels.n);
124      int x, y;
125      for(int i=0;i<photomaton->pixels.m;i++){
126          for(int j=0; j<photomaton->pixels.n;j++){
127              if(j%2==0){
128                  y=j-j/2;
129              }
130              else if(j%2!=0){
131                  y=j+photomaton->pixels.n/2-j/2-1;
132              }
133              if(i%2==0){
134                  x=i-i/2;
135              }
136              else if(i%2!=0){
137                  x=photomaton->pixels.m/2+i-i/2-1;
138              }
139              photomaton->pixels.data[x][y]=orig->pixels.data[i][j];
140          }
141      }
142      return success;
143  }
```

Algo et code : Matrice de convolution

- Filtre à pixel
- Gestion des out of bound



Algo et code : Matrice de convolution

```
155  pgm_error pgm_conv(pgm *conv, const pgm *const orig, const matrix *const kernel){
156      if(NULL== orig || NULL==conv || NULL==kernel){
157          return failure;
158      }
159      conv->max=orig->max;
160      matrix_alloc(&(conv->pixels), orig->pixels.m, orig->pixels.n);
161      int kernel_sum=matrix_sum(*kernel);
162      if(kernel_sum<1) kernel_sum=1;
163      for(int i=0;i<orig->pixels.m;i++){
164          for(int j=0; j<orig->pixels.n;j++){
165              double filter_sum=0.0;
166
167              for(int x=0; x<kernel->m;x++){
168                  for(int y=0;y<kernel->n;y++){
169                      int xi= i - kernel->m/2 + x;
170                      int yj= j - kernel->n/2 + y;
171                      if(xi>=0 && xi<conv->pixels.m && yj>=0 && yj<conv->pixels.n){
172                          filter_sum+=orig->pixels.data[xi][yj]*kernel->data[x][y]/kernel_sum;
173                      }
174                  }
175              }
176              if(filter_sum<0) filter_sum=0;
177              if(filter_sum>conv->max)filter_sum=conv->max;
178              conv->pixels.data[i][j]= (int)filter_sum;
179          }
180      }
181      return success;
182  }
```

Conclusion

- Evolutivité
- Facilité d’ajout de filtres
- Optimisation du code
- Arguments au programme

Questions