# MI01 : TP07 : Traitement d’une image (deuxième partie).

## Introduction

## Principe de la détection

## Algorithme de traitement

## Code

On ne réécrira que la partie ajoutée pour ce TP.

|  |  |
| --- | --- |
| mov esi, [ebp + 20]  mov edi, [ebp + 24]  mov ecx, [ebp+12]  sub ecx, 2  mov eax, [ebp+8]  lea edi, [edi + eax\*4 +4]  shl ecx, 16  push ebp  mov ebp, eax  ligne:  add cx, bp  sub cx, 2  colonne:  mov eax, [esi]  mov edx, eax  neg eax  mov ebx, eax    mov eax, [esi+4]  imul eax, 2  add edx, eax    mov eax, [esi+8]  add ebx, eax  add edx, eax    mov eax, [esi + 4\*ebp]  imul eax, 2  neg eax  add ebx, eax    mov eax, [esi + 4\*ebp + 8]  imul eax, 2  add ebx, eax    mov eax, [esi + 8\*ebp]  neg eax  add ebx, eax  add edx, eax    mov eax, [esi + 8\*ebp + 4]  imul eax, 2  neg eax  add edx, eax    mov eax, [esi + 8\*ebp + 8]  add ebx, eax  neg eax  add edx, eax    cmp ebx, 0  JGE positifb  neg ebx  positifb:  cmp edx, 0  JGE positifd  neg edx  positifd:  mov eax, ebx  add eax, edx  sub eax, 255  neg eax  mov edx, eax  shl eax, 8  add eax, edx  shl eax, 8  add eax, edx  mov [edi], eax    add esi, 4  add edi, 4  dec cx  test cx, 0FFFFh  jnz colonne  add esi, 8  add edi, 8  sub ecx, 000010000h  test ecx, 0FFFF0000h  jnz ligne        pop ebp  fin | adresse premier pixel tampon 1  adresse premier pixel tampon 2  ecx = hauteur  ecx = hauteur-2 (pour qu’on ne parcourt que la partie utile de l’image)  eax = largeur  pointe sur le premier pixel de sortie (tampon 2) où on stocke  partie haute ecx : compteur i, *i.e.* hauteur-2  Sauvegarde de ebp avant utilisation  ebp = eax = largeur  partie basse ecx : largeur  cx : compteur j, *i.e.* largeur-2  On va parcourir les pixels un par un. Pour chaque pixel, on applique le masque, et à chaque pixel du masque, un par un, on applique l’opérateur de Sobel.  Pixel du masque n°1 dans eax  On commence par mettre dans edx (qui remplit le rôle de Gy) le résultat de l’application de l’opérateur de Sobel (pour le 1° pixel : \*1)  On fait de même pour ebx (qui remplit le rôle de Gx) (application opérateur de Sobel pour le 1° pixel : \*(-1))  Pixel du masque n°2 dans eax  Maintenant, on additionne dans ebx ou edx  Pixel du masque n°3  Pixel du masque n°4  Pixel du masque n°5  Pixel du masque n°7  Pixel du masque n°8  Pixel du masque n°9  Si ebx position, on saute  Sinon, on le rend positif (valeur absolue)  De même pour edx  Une fois que ebx et edx sont positifs  On fait la somme |Gx|+|Gy| dans eax.  eax = 255-G (inversion du noir et du blanc)  On stocke ce résultat dans edx  On a mis le résultat dans chaque octet de eax pour avoir du noir et blanc (et pas juste un niveau d’intensité du bleu)  A la fin du traitement d’un pixel, on ajoute 4 à edi et esi pour passer à la colonne suivante  On décrémente le compteur colonnes (j) : s’il est égal à 0, on ne boucle plus (on ne fait pas le saut)  A la fin du parcourt d’une ligne de l’image, on ajoute 8 à edi et esi (en plus des 4 ajoutés précédemment) pour passer à la ligne suivante.  On décrémente le compteur lignes (i) : s’il est égal à 0, le programme est fini. |

## Conclusion