INF401

TP8et9 «Procédures, fonctions et paramètres, liens entre ARM et C »

Shaghayegh HAJMOHAMMADKASHI , Kimia KHADEMLOU– MIN3

8.1 Objectifs du TP

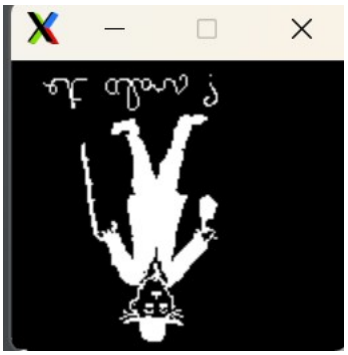
Le TP consiste à utiliser des algorithmes de traitement d'images pour appliquer divers effets sur des fichiers bitmap. Les tâches principales impliquent la création de fonctions en langage d'assemblage ARM qui effectuent différents traitements sur des images bitmap.

8.2 Découverte des transformations écrites en C

on a exécuté le affiché l'image
avec differents traitements :

option nv :

option nh :



option h :

option v :

option nhv :

option n :

option hv :

8.3 Négatif d’une image

Voici le code qui fait le négatif d’une image :

```
neg_image:
    mov r1, #0              @ Initialiser l'indice de ligne à 0
boucle:
    cmp r1, #HEIGHT         @ Comparer l'indice de ligne avec la hauteur de l'image
    bgt finboucle           @ Sortir de la boucle si nous avons atteint la fin de l'image

    mov r2, #0              @ Initialiser l'indice de colonne à 0
boucle2:
    cmp r2, #BYTES_PER_LINE @ Comparer l'indice de colonne avec le nombre d'octets par ligne
    bgt finboucle2          @ Sortir de la boucle si nous avons atteint la fin de la ligne

    ldrb r4, [r0]           @ Charger l'octet actuel de l'image dans r4
    mvn r4, r4              @ Inverser tous les bits de l'octet

    strb r4, [r0], #1       @ Stocker l'octet inversé dans l'image et avancer l'adresse
    add r2, r2, #1          @ Incrémenter l'indice de colonne
    b boucle2               @ Boucler si nécessaire

finboucle2:
    add r1, r1, #1          @ Incrémenter l'indice de ligne
    b boucle                @ Boucler si nécessaire

finboucle:
    mov pc, lr              @ Retourner
```

résultat d’éxecution :

8.4 Symétries d’une image

on a complété le fichier de symetrie_asm.S.
(vous trouvez le code de symétrie dans le fichier de code source)

résultat d’exécution :

option v :

option h :

exécutez avec la commande « arm-eabi-run transformer option » pour voir les autres manières de traitements.

8.5 Affichage de contenu

En se basant sue le fichier de « Affichage_contenu.c », on a écrit la version ARM de ce code.
Voici le script du code :

```
afficher_contenu:
__@ Function supports interworking.
__@ args = 0, pretend = 0, frame = 16
__stmfd   sp!, {fp, lr}
__add fp, sp, #4
__sub sp, sp, #16
__str r0, [fp, #-16]
__ldr r0, .L6
```

```
    mov r1, #168
    bl  afficher_entier
    ldr r0, .L6+4
    mov r1, #145
    bl  afficher_entier
    ldr r0, .L6+8
    ldr r1, .L6+12
    bl  afficher_chaine
    mov r3, #0
    str r3, [fp, #-8]
    mov r3, #12
    str r3, [fp, #-12]
    b   .L2
.L5:
    ldr r3, [fp, #-12]
    sub r3, r3, #1
    str r3, [fp, #-12]
    ldr r3, [fp, #-16]
    ldrb    r3, [r3, #0]    @ zero_extendqisi2
    ldr r0, .L6+16
    mov r1, r3
    bl  afficher_entier
    ldr r2, [fp, #-8]
    mov r3, #3040
    add r3, r3, #4
    cmp r2, r3
    beq .L3
    ldr r0, .L6+20
    ldr r1, .L6+12
    bl  afficher_chaine
.L3:
    ldr r3, [fp, #-12]
    cmp r3, #0
    bne .L4
    ldr r0, .L6+24
    ldr r1, .L6+12
    bl  afficher_chaine
    mov r3, #12
    str r3, [fp, #-12]
.L4:
    ldr r3, [fp, #-16]
    add r3, r3, #1
    str r3, [fp, #-16]
    ldr r3, [fp, #-8]
    add r3, r3, #1
    str r3, [fp, #-8]
.L2:
    ldr r2, [fp, #-8]
    mov r3, #3040
    add r3, r3, #4
    cmp r2, r3
    ble .L5
    ldr r0, .L6+28
    ldr r1, .L6+12
    bl  afficher_chaine
```

```
    sub sp, fp, #4
    ldmfd   sp!, {fp, lr}
    bx  lr
.L7:
    .align  2
.L6:
    .word   .LC0
    .word   .LC1
    .word   .LC2
    .word   .LC3
    .word   .LC4
    .word   .LC5
    .word   .LC6
    .word   .LC7
    .size   afficher_contenu, .-afficher_contenu
    .ident  "GCC: (GNU) 4.5.3"
```

conclusion :

En suivant les étapes du TP et en mettant en œuvre les fonctions nécessaires, nous devrions être capables de traiter des images bitmap pour obtenir les effets souhaités. Les procédures ARM et les fonctions C nous offrent la possibilité d'explorer différentes manipulations d'images tout en maintenant des performances optimales.