STI
TP N°1 : Morphologie Mathématique : application à la détection de mouvement

Master 1
Ghiles Mostafaoui

1 Introduction

1.1 Objectifs du TP

L'objectif de ce premier TP est de vous permettre de travailler et tester les opérateurs basiques en morphologie mathématique appliquée à la détection de mouvement par différence d'images. Vous appréhenderez les notions suivantes :

- la détection de mouvement par différence d'images consécutives
- le calcul d'une image de référence (fond fixe) par moyennes temporelles ou par filtrage médian
- la détection de mouvement par différence par rapport à l'image de référence
- l'érosion appliquée à des images binaires
- la dilatation appliquée à des images binaires
- L'ouverture et la fermeture appliquées à des images binaires

1.2 La bibliothèque NRC(Numerical Recipes in C)

Comme pour les TPs de Traitement d'images du 1er semestre, nous utiliserons dans le cadre des travaux pratiques de ce module la bibliothèque Numerical Recipes (NRC) qui contient un ensemble de fonctions qui permettent :

- de gérer les entrées/sorties : lire et écrire des images de type PGM ou PPM (voir le fichier nrio.c)
- de gérer les allocations mémoire : allouer en mémoire des matrices de différents types et de différentes taille (voir le fichier nralloc.c)
- de réaliser des calculs vectoriels et matriciels (voir le fichier nrarith.c)

Pour utiliser cette bibliothèque il suffit d'inclure dans votre fichier ".c" les différents Headers : "def.h", "nrio.h", "nralloc.h" et "nrarith.h".
Nous utiliseront principalement les fonctions suivantes :

- **byte ** LoadPGM_bmatrix(char *filename, long *nrl, long *nrh, long *ncl, long *nch)** : Lecture d'une image de type PGM
- **rgb8 ** LoadPPM_rgb8matrix(char *filename, long *nrl, long *nrh, long *ncl, long *nch)** : Lecture d'une image de type PPM
- **void SavePGM_bmatrix(byte **m, long nrl, long nrh, long ncl, long nch, char *filename)** : Ecriture d'une image de type PGM
- **void SavePPM_rgb8matrix(rgb8 **m, long nrl, long nrh, long ncl, long nch, char *filename)** : Ecriture d'une image de type PPM
- **byte** bmatrix(long nrl, long nrh, long ncl, long nch)** : Allocation d'une matrice de "byte"
- **rbg8** rgb8matrix(long nrl, long nrh, long ncl, long nch)** : Allocation d'une matrice de "rgb8"
- **int** imatrix(long nrl, long nrh, long ncl, long nch)** : Allocation d'une matrice de "int"

- **void free_bmatrix(byte **m, long nrl, long nrh, long ncl, long nch)** : libération de la mémoire pour une matrice de type "byte"
- **vois free_rgb8matrix(rgb8 **m, long nrl, long nrh, long ncl, long nch)** : libération de la mémoire pour une matrice de type "rgb8"
- **void free_imatrix(int **m, long nrl, long nrh, long ncl, long nch)** : libération de la mémoire pour une matrice de type "int"

2 Prise en main : Lecture et écriture d'une image de type PGM ou PPM

Un fichier "exemple.c" est fourni pour rappeler comment lire et écrire des images de type pgm ou ppm. On rappelle que pour compiler votre programme il suffit d'inclure tous les .c comme suit :

gcc -o tp1 tp1.c nrio.c nralloc.c nrarith.c

Note : Pensez à bien libérer vos mémoires !

3 Morphologie Mathématique : Erosion, Dilatation, Ouverture et Fermeture

Programmez les 2 opérateurs de bases en morphologie mathématique que sont l'érosion et la dilatation. Testez votre programme sur l'images 'carre.pgm'.

Une fois ces deux opérateurs testés et validés, utilisez les pour réaliser les ouvertures et fermetures pour :

- combler le trou au milieu du carré dans l'image 'carreTrou.pgm'
- supprimer le bruit (le petit carré !!) dans l'image 'carreBruit.pgm'

4 Détection de mouvement par différence d'images consécutives

Deux séquences d'images vous seront fournies en format ppm : Fomd ("Fight On Men Down") et Lbox ("Left Box"). Le premier travail à réaliser est de détecter les pixels en mouvement sur chacune des images des 2 séquences en utilisant une simple différence seuillée d'images successives ($I(t) - I(t-1)$). Pensez à sauvegarder les images "binaires" résultantes dans un répertoire différent que vous nommerez "Resultats". Il faut aussi pensez à tester plusieurs seuils afin de maximiser le rapport signal/bruit.

Une fois les 2 séquences traitées, utiliser les opérations programmées précédemment (ouverture, fermeture) pour améliorer le résultat de la détection (comme vu en cours). Pensez là aussi à sauvegarder vos résultats dans un autre répertoire nommé "ResultatsMorpho".

5 Extraction d'une image de Référence

Calculez pour chacune des 2 séquences une image de référence (fond fixe) en utilisant les deux méthodes suivantes :

- une moyenne temporelle
- et un filtre médian

Comparez les 2 résultats !!

6 Détection de mouvement avec image de référence

Réalisez de nouveau une détection de mouvement pour les 2 séquences en utilisant cette fois une différence avec l'image de référence calculée précédemment ($I(t) - IRef$).

Optimisez les résultats avec les opérations de morphologie mathématique.

7 Etiquettage et Caractérisation des formes binaires

A l'aide de l'algorithme vu en en cours de TSI, étiqueter les images binaires qui résultent de la détection d'images. Pour chaque image, sauvegarder dans un fichier .txt les données suivantes :

- nombre de régions
- pour chaque région :
  - numéro de la région (Label ou étiquette)
  - la taille
  - les coordonnées du centre de gravité
  - les écarts types $\sigma_x^2, \sigma_y^2, \sigma_{xy}$
  - la direction principale (en degrés)
  - la moyenne des niveaux de gris
  - les moyennes sur les composantes R, G et B
  - l'histogramme des niveaux de gris