Université de Paris Sud

Master 2 Pro IST

TiVi

Détection de mouvement

1 Introduction

Le but de ce TP est de de construire deux chaines de traitement pour la détection de mouvement. Ce TP se base sur le codage des opérateurs des TP précédents. La première chaine de traitement est une simple différence d'image (*Frame Difference*) avec seuil fixe tandis que la seconde est basé sur le filtre adaptatif $\sigma\Delta$ à seuils adaptatifs.

Les codes C fournis sont séparés en modules :
– `def.h` : ajout de quelques modifications (par rapport au TP précédent)
– `sequence` : gestion du nom et numéro de séquence
– `parser` : pour lire le fichier de configuration
– `motionNR` : opérateurs de morphologie mathématique
– `test_motionNR` : fonctions de test des opérateurs de détection de mouvement

Faire attention en récupérant le zip de ne pas écraser les anciens codes...
Dans un premier temps, le but est de coder les deux opérateurs (FD et $\Sigma\Delta$ sans pré-traitement ni post-traitement avec d'évaluer leurs performances. Dans un second temps, vous devrez d'abord ajouter un post-traitement et évaluer sont effet sur les images binaires produites, puis ajouter un pré-traitement et estimer son intérêt.
Vos explications seront illustrés par des exemples de résultats

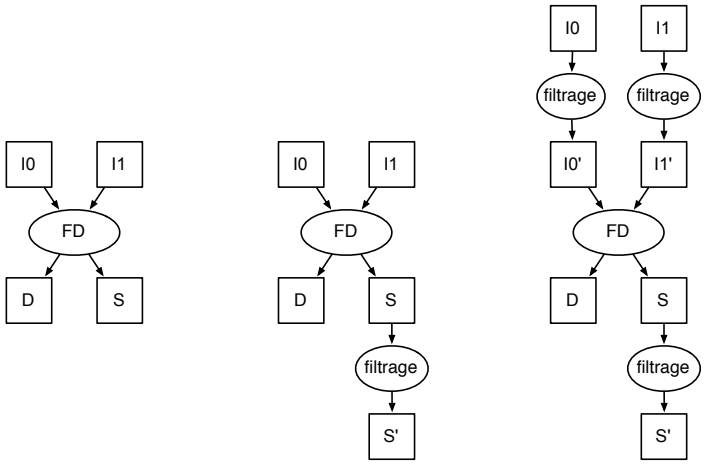

FIGURE 1 – *Frame Différence* : (1) FD, (2) FD+post-traitement, (3) FD + post-traitement + pré-traitement

2 Travail à faire

1. ajouter les codes d'opérateurs manquant dans le fichier `motionNR`.
2. completer les fonctions de tests (pour ajouter un post-traitement puis un pré-traitement) dans le fichier `test_motionNR`.

2.1 Frame-Difference

Algorithm 1: Frame Difference

foreach *pixel* x **do** [step #1 : D_t computation]
  $D_t(x) = |I_t(x) - I_{t-1}(x)|$
foreach *pixel* x **do** [step #2 : D_t threshold]
  if $D_t(x) < \theta$ **then** $S_t(x) \leftarrow 0$ **else** $S_t(x) \leftarrow 1$

1. Coder l'opérateur de différence d'image FD (algo. 1) et le tester sur la séquence car3. Que peut on observer ?
2. Ajouter un post-traitement morphologique, puis un pré-traitement morphologique. Conclure.

2.2 Sigma-Delta

Algorithm 2: Basical $\Sigma\Delta$

foreach *pixel* x **do** [step #1 : M_t estimation]
  if $M_{t-1}(x) < I_t(x)$ **then** $M_t(x) \leftarrow M_{t-1}(x) + 1$
  if $M_{t-1}(x) > I_t(x)$ **then** $M_t(x) \leftarrow M_{t-1}(x) - 1$
  otherwise $M_t(x) \leftarrow M_{t-1}(x)$
foreach *pixel* x **do** [step #2 : O_t computation]
  $O_t(x) = |M_t(x) - I_t(x)|$
foreach *pixel* x **do** [step #3 : V_t update]
  if $V_{t-1}(x) < N \times O_t(x)$ **then** $V_t(x) \leftarrow V_{t-1}(x) + 1$
  if $V_{t-1}(x) > N \times O_t(x)$ **then** $V_t(x) \leftarrow V_{t-1}(x) - 1$
  otherwise $V_t(x) \leftarrow V_{t-1}(x)$
foreach *pixel* x **do** [step #4 : $\hat{E}_t$ estimation]
  if $O_t(x) < V_t(x)$ **then** $\hat{E}_t(x) \leftarrow 0$ **else** $\hat{E}_t(x) \leftarrow 1$

1. Coder l'opérateur $\Sigma\Delta$ (algo. 2) et le tester sur la séquence car3. Que peut on observer ?
2. Ajouter un post-traitement morphologique, puis un pré-traitement morphologique. Conclure.

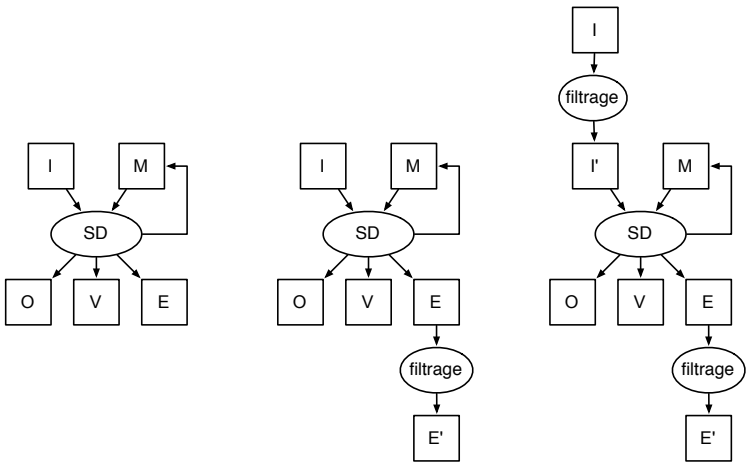

FIGURE 2 – *Sigma-Delta* : (1) $\Sigma\Delta$, (2) $\Sigma\Delta$+post-traitement, (3) $\Sigma\Delta$ + post-traitement + pré-traitement