TP-Filtrage Particulaire - Diverses applications
Télécom SudParis - MAT4501

1 Introduction

Le but de ce TP est d'utiliser les techniques de filtrage particulaire sur diverses applications : économie, poursuite sur image, poursuite radar... Les manipulations qui suivent sont à préparer **avant le TP** : créer un répertoire *TP_filtrage_particulaire* contenant les dossiers *exerice1*, *exercice2*. Dans le répertoire *exercice2*, placez les différentes séquences d'images que vous aurez téléchargées et décompressées dans des répertoires *sequence1*, *sequence2*... Toujours dans *exercice2*, télécharger les divers fichiers Matlab (*calcul_histogramme.m*, *lecture_image.m*...). Ces fichiers sont également disponibles pour une implémentation en Python. Tous ces fichiers se trouvent à l'adresse suivante : `http://www-public.it-sudparis.eu/~petetin/cours.html`, rubrique Filtrage Particulaire.

2 Modèle

On cherche dans ce TP à estimer à chaque instant une fonction d'une variable aléatoire dite cachée, X_n, à partir d'une suite de variables aléatoires appelées observations, $Y_{0:n} := (Y_0, Y_1 \cdots Y_n)$, fonctions des variables cachées pour tout n. Le processus $(X_{0:n}, Y_{0:n})$ est dit chaîne de Markov cachée, si la loi jointe du couple $(X_{0:n}, Y_{0:n})$ de réalisation $(x_{0:n}, y_{0:n})$ se factorise sous la forme suivante :

$$p(x_{0:n}, y_{0:n}) = p(x_0) \prod_{i=1}^{n} f_{i|i-1}(x_i|x_{i-1}) \prod_{i=0}^{n} g_i(y_i|x_i).$$

Pour des modèles à bruit additif, on se donne une équation d'état ainsi qu'une équation observation :

$$\begin{aligned} X_n &= f(X_{n-1}) + U_n \quad (1) \\ Y_n &= g(X_n) + V_n \quad (2) \end{aligned}$$

où les U_i, V_i sont de lois connues, les U_i indépendantes entre eux et indépendantes des V_j (elles aussi indépendantes entre elles). On en déduit ainsi la loi de X_i, conditionnellement à la variable X_{i-1}, ainsi que celle de Y_j, conditionnellement à X_j, ce qui caractérise donc parfaitement la loi du couple $(X_{0:n}, Y_{0:n})$.

3 Exercice 1 : un petit échauffement...le modèle de Kitagawa

Ce modèle sert souvent d'indicateur de performance pour comparer différents algorithmes de filtrage et est utilisé en économétrie. Le modèle est décrit par les deux équations suivantes :

$$\begin{cases} X_n &= 0.5X_{n-1} + 25\frac{X_{n-1}}{1+X_{n-1}^2} + 8\cos(1.2(n)) + U_n \\ Y_n &= \frac{X_n^2}{20} + V_n \end{cases}$$

où $U_n \sim \mathcal{N}(0, Q)$ et $V_n \sim \mathcal{N}(0, R)$. Tous les bruits sont indépendants entre eux. On cherche à estimer la variable X_n à chaque instant. Les paramètres à initialiser sont les suivants : $N = 50$ (nombre de particules), $Q = 10$, $R = 1$, $T = 50$ (longueur du scénario).

1. En examinant l'équation d'observation, pourquoi semble-t-il difficile de déduire l'état caché à partir de l'observation seule ?

2. Expliciter la loi de transition $f_{n|n-1}(x_n|x_{n-1})$ ainsi que la vraisemblance $g_n(y_n|x_n)$.

3. A partir de l'équation d'état, créer une fonction (à vous de déterminer ce qu'il faut mettre en entrée et en sortie) afin de générer une trajectoire, $x_{0:T}$, de longueur T.

4. Créer une fonction qui retourne un jeu d'observation (selon le modèle ci-dessus) à partir d'une trajectoire donnée. N'oubliez pas de créer un programme *principal.m* dans lequel vous empilez vos fonctions au fur et à mesure.

5. On suppose à présent qu'on ne dispose que des observations, qui arrivent au fur et à mesure, et on cherche à chaque instant un estimateur de X_n. Créer une fonction *filtrage_particulaire.m* qui prend en arguments un jeu de particules et les poids associés, les paramètres de bruit du modèle et qui retourne un nouveau jeu de particules et de poids, ainsi que l'estimateur de l'état caché estimé :
   - Tirer les nouvelles particules suivant la loi de transition $f_{n|n-1}(x_n|x_{n-1})$.
   - Calculer la vraisemblance de chaque particule, à l'aide de la loi $g_n(y_n|x_n)$.
   - En déduire le poids normalisé de chaque particule.
   - Donner un estimateur de l'état caché.
   - Réechantillonner les particules. On créera ou utilisera pour cela une fonction *reechantillonnage.m* .
6. Calculer et afficher l'erreur quadratique entre la vraie trajectoire et la trajectoire estimée.

7. Tracer la vraie trajectoire, la trajectoire observée et la trajectoire estimée, de différentes couleurs .

8. Faire varier le nombre de particules et commenter (erreur quadratique, temps d'exécution...). Pour afficher le temps d'exécution d'un programme on pourra utiliser les fonctions *tic* (au début du programme) et *toc* (à la fin du programme).

4 Exercice 2 : Suivi de visage sur une séquence vidéo

Le but de cet exercice est d'utiliser le filtrage particulaire sur une séquence vidéo afin de suivre des objets. On représentera l'objet d'intérêt par un rectangle, dont on supposera que les dimensions sont fixes au cours du temps. Le vecteur aléatoire X_n, représentant l'état caché, est donc un vecteur à deux dimensions, $X_n = [X_1, X_2]_n^T$ où X_1 et X_2 représentent respectivement l'abscisse et l'ordonnée du point haut/gauche du rectangle. On cherchera donc à chaque instant la position de l'objet traqué. Voici les codes dont vous disposez pour examiner les caractéristiques des images :

- *lecture_image* : permet de lire une séquence d'image dans le dossier spécifié dans ce fichier. Elle retourne la première image du répertoire, le nom de toutes les images, le nombre d'image ainsi que le nom du dossier.
- *selectionner_zone* : permet de sélectionner une zone sur l'image et retourne les caractéristiques du rectangle sélectionné : abscisse et ordonnée du point haut/gauche du rectangle, longueur et largeur.
- *calcul_histogramme* : prend en argument une image, les caractéristiques d'un rectangle ainsi que le nombre de couleurs représentatives N_b OU une carte de couleur existante, et retourne l'image sélectionnée, la carte de couleurs associée à N_b, ainsi que l'histogramme de couleur de la zone sélectionnée.

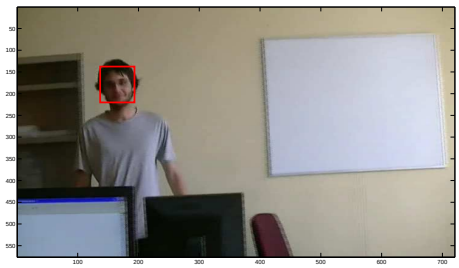

FIGURE 1 – Un 1A de 2011, à ses heures perdues

L'équation dynamique est donnée par la relation suivante :

$$X_n = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times X_{n-1} + U_n \tag{3}$$

où $U_n \sim \mathcal{N}(0_{2\times 2}, \begin{pmatrix} C_1 & 0 \\ 0 & C_2 \end{pmatrix})$.

On s'intéresse à présent à l'expression de la vraisemblance $g_n(y_n|x_n)$. Pour cela, décrivons la méthode employée pour faire le suivi d'image.

Nous allons commencer par définir une zone de l'image que l'on souhaite suivre durant la séquence. On va alors calculer un histogramme de couleur associée à cette zone, à l'aide la fonction *calcul_histogramme.m*. Plus précisément, pour un paramètre N_b donné, cette fonction retourne un vecteur q de taille N_b, tel que $q(n)$ indique le nombre de pixel étant de la n-ième couleur (l'espace RGB est divisé en N_b couleurs).

A un instant k, chaque particule x_k^i va définir une zone sur l'image. On aura alors pour chaque particule un histogramme de couleur associé, calculée de la même manière que précédemment. Il sera alors possible de comparer l'histogramme de référence (on supposera que l'histogramme de couleur de l'objet traqué est invariant dans le temps) avec un histogramme d'une particule x_k^i. Plus précisément, on définit la distance suivante entre deux histogrammes de couleur q et q' :

$$D(q, q') = \left(1 - \sum_{i=1}^{N_b} \sqrt{q(n)q'(n)}\right)^{\frac{1}{2}}.$$

On introduit enfin la loi de probabilité suivante pour la vraisemblance :

$$g_n(y_n|x_n) \propto \exp\{-\lambda D^2(q, q'(x_n))\},$$

où q est l'histogramme de référence et $q'(x_k)$ l'histogramme défini par le vecteur d'état x_k. La densité $p(y_k|x_k)$ n'a besoin d'être connue qu'à une constante près car les poids sont normalisés. Les différentes étapes à suivre pour mettre en oeuvre le pistage sont les suivantes :

- Initialiser les différents paramètres. On prendra $N = 50$ particules, pour commencer, $N_b = 10$, $\lambda = 20$, $C_1 = C_2 = 300$.
- Afficher la première image de la séquence à l'aide de la fonction *lecture_image.m* puis appeler la fonction *selectionner_zone* pour sélectionner la zone de l'image à pister. Cette fonction retourne un vecteur 1×4, qui contient l'abscisse et l'ordonnée du point haut/gauche du rectangle, ainsi que la longueur et la largeur.
- Initialiser les particules selon une loi Normale, autour du point haut/gauche, avec un écart-type égal à $\sqrt{300}$.
- A présent, les images nous arrivent une par une. A chaque instant, on acquiert une nouvelle image, à l'aide de la fonction *imread*.
- On effectue le filtrage particulaire : on commence par propager les particules (tirage suivant la loi de transition). Pour chaque particule, on calcule les caractéristiques du rectangle associé (coordonnée du point haut/gauche, longueur, largeur). Pour rappel, les longueurs et largeurs sont identiques à celles que l'on a initialisées au départ. On calcule alors l'histogramme de couleur associé au rectangle, on calcule sa distance avec l'histogramme de référence, puis on en déduit le poids de la particule considérée.
- Après normalisation des poids, on réechantillonne les particules.
- En déduire la position estimée du point haut/gauche du rectangle qui piste l'objet désiré.
- Afficher l'image, tracer le rectangle estimé (fonction *rectangle*), et représenter la position des particules sur l'image par des croix. Rajouter la commande *pause* afin de pouvoir admirer le résultat. Vous pouvez éventuellement sauvegarder les positions estimées dans un vecteur afin de tracer à chaque instant la trajectoire de l'objet pisté.

Questions

1. Le modèle d'état utilisé vous semble-t-il pertinent ? Pourrions-nous l'améliorer ?

2. Que se passe-t-il lorsque le mouvement de l'objet pisté est important (forte accélération) ? Pourquoi ? Comment résoudre le problème ?

3. Lorsque l'on augmente ou diminue le paramètre λ, que constatez-vous sur le nombre de particules affichées ? Pourquoi ? Il est possible de mesurer ce phénomène en évaluant le paramètre $N_{eff}(n) := \frac{1}{\sum_{i=1}^{N}(w_n^i)^2}$, où les w_n^i sont les poids des particules *avant* réechantillonnage, à l'instant n. Tracer la fonction $N_{eff}(n)$, pour $n = 1 \cdots T$, où T est la longueur de la séquence, pour deux valeurs différentes de λ.

4. Analyser le comportement de l'algorithme lors des croisements, en fonction des divers paramètres de l'algorithme.

5. Quelle est la différence entre un algorithme de reconnaissance implémenté dans un appareil photo numérique et l'algorithme que vous avez implémenté ?

Nous avons supposé que la dimension de l'objet pisté était invariante au cours du temps. Ceci ne permettra pas de pister correctement l'objet si ce dernier s'approche ou s'éloigne de la caméra. On complète alors le vecteur d'état $X_n = [X_1, X_2, X_3]_n^T$, où X_3 est un paramètre d'échelle, par rapport à la dimension initiale. Cette composante sera initialisée selon une loi normale de moyenne 1 et de variance faible. La matrice devient $u_n \sim \mathcal{N}(0_{3\times 3}, \begin{pmatrix} C_1 & 0 & 0 \\ 0 & C_2 & 0 \\ 0 & 0 & C_3 \end{pmatrix})$, avec $C_3 = 2/100$ dans un premier temps. Compléter l'algorithme de façon à ce que la taille du rectangle s'adapte aux effet de zoom. Il suffit pour cela de multiplier la taille initiale du rectangle par la moyenne des paramètres d'échelles de chaque particule.