Filtre particulaire parallèle

V. Teulière et O. Brun

LAAS-CNRS
7, avenue du Colonel Roche, 31077 Toulouse Cedex, France.
e-mail: teuliere,brun@laas.fr

Abstract— **Particle filtering may cope with nonlinear and/or non-Gaussian models. As a counterpart, this technique suffers from an heavy computation cost and can not always satisfy real time constraints. A data parallel algorithm is proposed to achieve real-time particle filtering.**
Le filtrage particulaire permet de traiter les modèles non-linéaires et/ou non gaussiens. En contrepartie, cette technique nécessite un coût algorithme élevé parfois imcoptabible avec le temps réel. Une approche paralléle est proposée pour mettre en oeuvre le filtrage particulaire en temps réel.

Keywords— **Particle filtering, parallel algorithm.**

I. Introduction

Le problème de filtrage Bayésien consister à estimer l'étar x_k d'un processus stochastique à partir d'une série de mesures issues d'un processus d'observation y_k; les relations entre les processus x_k et y_k étant connues. Le processus x_k décrit l'évolution des variables d'état du système et y_k est une mesure bruitée du processus x_k. L'estimateur optimal n'est réalisable en dimension finie que dans le cas des systèmes linéaires à bruits gaussiens,c'est le filtre de Kalman [10].

Dans le cas général où le système est non-linéaire et/ou à bruits non-gaussiens, le filtrage particulaire (FP) constitue une solution performante. Ce type d'algorithme fut initialement introduit pour le GdR traitement du signal [7] D'autres équipes ont développé des techniques apparentées, c'est le cas par exemple du "Bootstrap Filter" [5]. L'idée consiste à représenter l'espace de probabilité de l'état par des mesures ponctuelles aléatoires de masses conditionnées par les observations.

L'avantage majeur de cette technique est qu'elle permet de prendre en compte tous les aspects dynamiques et non-linéaires sans aucunes restrictions ni approximations, tout en assurant l'optimalité du traitement. Ainsi, le FP a clairement démontré sa supériorité sur les approches classiques, en simulation ou même sur données réelles, dans de de nombreux domaines : RADAR [8] , LORAN C [9] et GPS [4]. Cependant, malgré ces succés indéniables, cette technique souffre d'un coût algorithmique souvent incompatible avec les impératifs du temps réel.

La mise en oeuvre temps réel du FP est toutefois possible en exploitant la structure parralléle inhérente à l'algorithme. Cet article présente la parallélisation du FP. Dans la section II, nous décrivons le problème d'estimation à temps dicret et rappelons l'algorithme de FP avec redistributions. L'algorithme parralléle est ensuite présentée au paragraphe III. Nous analysons ensuite les peformances obtenues sur le problème de trajectographie passive SONAR pour cibles manoeuvrantes dans la partie IV.

II. Filtrage particulaire

A. Le problème de filtrage à temps discret

Nous rappelons ici le problème d'estimation à temps discret. Soit x_k un processus stochastique à réalisation Markovienne dont la dynamique est régie par :

$$x_{k+1} = f_k(x_k, w_k) \tag{1}$$

où f_k est la fonction de transition et w_k le bruit de dynamique. A des instants discrets k, des mesures y_k sont disponibles et liées à l'état par l'équation d'observation :

$$y_k = h_k(x_k, v_k) \tag{2}$$

où h_k est la fonction de mesure et v_k le bruit d'observation. Les densités de probabilité des bruits de dynamique $p(w_k)$, d'observation $p(v_k)$ et de l'état initial $p(x_0)$ sont supposées connues. L'estimateur optimal au sens du minimum de variance coïncide avec l'espérance conditionnelle $E[x_k|Y_k = y_1, \ldots, y_k]$ qui dépend de la densité de probabilité de conditionnelle de l'état au vues des observations $p(x_k|Y_k)$. L'évolution de cette densité conditionnelle est régie par les lois de transition $p(x_k|x_{k-1})$ (Chapman-Kolmogorov) et de mesure $p(y_k|x_k)$ (Bayes). Elle se calcule récursivement à partir de l'équation suivante :

$$p(x_k|Y_k) = \frac{p(y_k|x_k) \int p(x_k|x_{k-1}) p(x_{k-1}|Y_{k-1}) dx_{k-1}}{p(y_k|Y_{k-1})} \tag{3}$$

L'équation (3) constitue la solution formelle au problème de filtrage à temps discret. Mais, comme il a été souligné dans l'introduction, sa résolution en dimension finie n'est possible que dans le cas très restrictif du cas linéaire Gaussien.

B. Filtrage Particulaire

Le FP permet de résoudre les équations de filtrage sans aucune restriction. Rigoureusement, les équations du filtrage peuvent être interprétées comme un processus générateur de naissances et de morts. L'objet étant ici la parallélisation de l'algorithme, nous nous limitons à rappeler l'algorithme basée sur un support ponctuel pondéré assortis de redistributions, pour plus de détails voir [3].

• **Initialisation**
La densité de probabilité initiale de l'état $p(x_0)$ est approché par un peigne de Dirac de N particules :

$$p(x_0) = \frac{1}{N}\sum_{i=1}^{N}\delta_{x_0^i}(x_0) \quad (4)$$

où δ_x désigne la mesure de Dirac de support x et N le nombre de particules. Le support particulaire initial $\{x_0^i\}_{i=1,...,N}$ est obtenu par tirage aléatoire conformément à $p(x_0)$.

• **Prédiction - Exploration a priori**
Chaque particule x_k^i évolue indépendamment selon le flot stochastique de l'équation dynamique (1) :

$$x_k^i = f_k(x_{k-1}^i, w_{k-1}^i) \quad (5)$$

par tirage aléatoire du bruit de dynamique w_{k-1}^i selon la densité de probabilité $p(w_{k-1})$.

• **Correction - pondération**
L'étape de correction consiste à utiliser la mesure y_k pour pondérer chacune des particules en accord avec le rapport Bayésien. Le poids ρ_k^i de chacune des particules (initialement égaux à $1/N$), se calcule récursivement par :

$$\rho_k^i = \frac{p(y_k|x_k^i)}{\sum_{j=1}^{N} p(y_k|x_k^j)\rho_{k-1}^j}\rho_{k-1}^i \quad (6)$$

Dans une majorité de systèmes, le bruit de mesure est additif Gaussien, i.e :

$$y_k = h_k(x_k) + v_k \text{ avec } E[v_k] = 0 \text{ et } E[v_k v_k^T] = R \quad (7)$$

la densité de probabilité de mesure s'exprime alors facilement :

$$p(y_k|x_k^i) = \Gamma_{y_k}\left(h\left(x_k^i\right), R_k\right) \quad (8)$$

où $\Gamma_x(\bar{x}, V)$ désigne une distribution Gaussienne de moyenne $\bar{x}$ et de variance V.

Ce support d'état discret et pondéré constitue l'approximation particulaire de la densité de l'état x_k au vues de la série de mesures jusqu'à l'instant courant $Y_k = y_1, \ldots, y_k$.

• **Calcul de l'estimé**
L'estimateur à minimum de variance $\widehat{x}_k$ n'est rien d'autre que la somme pondéré des états particulaires x_k^i:

$$\widehat{x}_k = \sum_{i=1}^{N}\rho_k^i x_k^i \quad (9)$$

Il est possible de calculer n'importe quel moment de la distribution conditionnelle, et notamment la covariance de l'erreur d'estimation $\widetilde{P}_k$ qui fournit un intervalle de confiance sur l'estimé :

$$\widetilde{P}_k = \sum_{i=1}^{N}\rho_k^i x_k^i {x_k^i}^T - \quad \widehat{x}_k\widehat{x}_k^T \quad (10)$$

• **Redistribution**
Ces deux étapes de prédiction et de correction forment la boucle principale de l'algorithme de FP. L'inconvénient majeur de cette approche brute réside dans le défaut de convergence uniforme. Avec l'accumulation des mesures, la variance des poids croît. En pratique, le poids se concentre rapidement sur une seule particule : la représentation de la densité est alors dégénérée. La solution régularisante consiste à redistribuer le support particulaire périodiquement selon la densité pré-acquise $p(x_k|Y_k)$. L'idée est d'éliminer les trajectoires peu vraisemblables au profit des trajectoires de poids élevé. Il existe de nombreuses techniques de redistribution. Dans le contexte du parallélisme, la redistribution proportionnelle est la plus adéquate. Chacune des particule i de poids $\rho_k^i > 1/(2N)$ est dupliquée en n_k^i particules selon :

$$n_k^i = PE\left[\rho_k^i N_c\right] \quad (11)$$

où PE désigne l'entier le plus proche et N_c le nombre de particules désirées après redistribution, soit le nombre de particules initial. Si le poids d'une particule est inférieur à $1/2N$, celle-ci meurt. Chacune des particules dupliquées $j = 1, \ldots, n_k^i$ se partage le poids de la particule initiale :

$$j = 1, \ldots, n_k^i \quad \rho_k^j = \frac{\rho_k^i}{n_k^i} \quad (12)$$

Cette méthode de redistribution est de complexité réduite. En contrepartie, le nombre total de particules varie autour du nombre initial $N_c = N$ qui joue ici le rôle d'une consigne. En pratique, cette variation n'excéde pas 5 %.

La redistribution n'est pas appliquée à chaque instant contrairement au "Bootstrap Filter" [5]. Pour N fini, la redistribution systématique peut avoir des conséquences désastreuses, pour n'en citer que deux : pertes de mode ou convergence vers un mode parasite. En effet, cette procédure induit une perte de diversité, il est préférable d'acquérir suffisamment d'information avant de la déclencher adaptativement à l'aide d'un indicateur heuristique de dégénérescence. Le nombre de particules efficaces [1]

$$N_{eff} \approx \frac{1}{\sum_{i=1}^{N}\left(\rho_k^i\right)^2} \quad (13)$$

fournit un bon indicateur de dégénérescence.

III. Algorithme de FP parallèle

L'algorithme de FP possède une structure intrinsèque parallèle. En effet, l'exploration aléatoire de l'espace d'état est régie par des entités indépendantes. Les particules interagissent uniquement lors de la normalisation des poids, c'est à dire lors du calcul de l'estimée et lors des redistributions. Durant les autres étapes les particules n'interagissent aucunement

La manière la plus judicieuse pour exploiter cette structure parallèle est d'utiliser une approche synchrone de type

parallèlisme de données [2]. Cela consiste à partitionner l'ensemble des N particules en plusieurs sous-ensembles indépendants de tailles égales. Chacun de ces sous-ensembles est alors affecté à un processeur.

Avant de décrire l'algorithme parallèle, introduisons quelques notations utilisées par la suite :

- $P+1$ est le nombre de processeurs.
- T_j désigne la tâche sur le processeur $j = 0 \dots P$.
- n_k^j est le nombre de particules assignées à la tâche T_j à l'instant k ($n_0^j \approx N/P$).
- $\left\{x_k^{i,j}, \rho_k^{i,j}\right\}_{i=1,\dots,n_j^k}$ est le support particulaire associé à la tâche T_j à l'instant k. $x_k^{i,j}$ est l'état de la particule et $\rho_k^{i,j}$ son poids.

La manière la plus performante pour calculer l'estimée est d'utiliser une approche parallèle maître/esclave. Ainsi, une seule tâche, appelée tâche maître T_0 est dédiée à ce calcul. Les autres tâches $T_{j=1\dots P}$, appelées tâches esclaves, effectuent la prédiction et à la pondération non normalisées de leur support particulaire $\left\{x_k^{i,j}, \rho_k^{i,j}\right\}_{i=1,\dots,n_j^k}$ en parallèle et indépendemment. Le point clé et la force de l'algorithme résident dans la minimisation des échanges entre tâches. Seul l'échange de variables scalaires est nécessaire au calcul de l'estimé $\widehat{x}_k$ et au contrôle de la redistribution. Ces variables sont calculées par chacune des tâches esclaves et constituent une agrégation d'informations locales. Le principe de cette approche est décrit sur la figure 1.

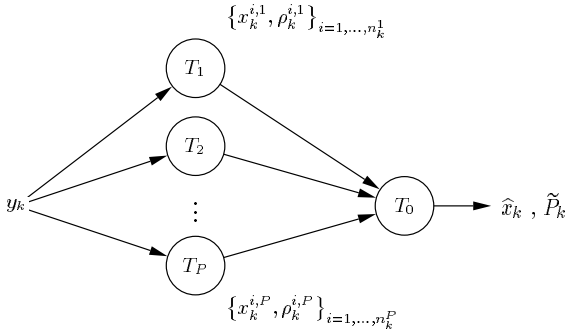


Fig. 1. Approche par parallélisme de données synchrone.

Nous allons examiner les deux points clés de l'algorithme de FP parallèle. Le premier point est détaillé au paragraphe III-A et constitue la boucle principale de l'algorithme : prédiction et pondération au niveau esclave, calcul de l'estimé et de l'erreur de covariance et contrôle de dégénérescence au niveau maître. En section III-B, nous examinons le problème des redistributions et d'équilibrage des charges de calcul entre processeurs.

A. Boucle principale

A l'instant initial $k = 0$, chacune des tâches esclaves $T_{j=1\dots P}$ initialise indépendemment son support particulaire $\left\{x_k^{i,j}, \rho_k^{i,j}\right\}_{i=1,\dots,n_j^k}$ selon $p(x_0)$.

Ensuite, chacune des tâches rentre dans la boucle principale. Nous détaillons à présent les calculs effectués à l'itération k par chacune des tâches esclaves $T_{j=1\dots P}$ et maître T_0. La figure 2 décrit cette boucle principale (les pointillés représentent les communications entre tâches).

A.1 Prédiction et pondération non normalisée

Chaque tâche esclave $T_{j=1\dots P}$ effectue la prédiction de son support particulaire selon le flot stochastique dynamique (1):

$$x_k^{i,j} = f_k(x_{k-1}^{i,j}, w_k^{i,j}) \tag{14}$$

A la réception de la mesure y_k , les poids non normalisés sont calculés par multiplication par le facteur Bayésien $p(x_k^i|y_k)$:

$$\rho_k^{i,j} = \exp\left[-\frac{1}{2}\left(y_k - h(x_k^{i,j})\right)^T R_k^{-1}\left(y_k - h(x_k^{i,j})\right)\right]\rho_{k-1}^{i,j} \tag{15}$$

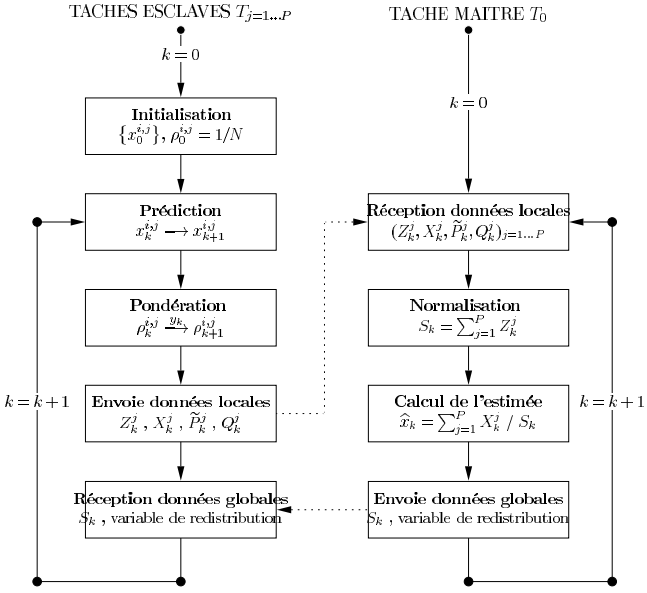


Fig. 2. Boucle principale.

Ensuite chaque tâche esclave $T_{j=1\dots P}$ procède aux calculs des variables suivantes :

$$Z_k^j = \sum_{i=1}^{n_k^j} \rho_k^{i,j} \quad , \quad X_k^j = \sum_{i=1}^{n_k^j} \rho_k^{i,j} x_k^{i,j}$$

$$\widetilde{P}_k^j = \sum_{i=1}^{n_k^j} \rho_k^{i,j} x_k^{i,j} x_k^{i,j^T} \quad \text{et} \quad Q_k^j = \sum_{i=1}^{n_k^j} \left(\rho_k^{i,j}\right)^2$$

Ces variables représentent une agrégation des données locales aux tâches esclaves. Z_k^j représente le poids non-normalisé de la tâche T_j et X_k^j son estimé local non-normalisé. $\widetilde{P}_k^j$ et Q_k^j sont utilisés respectivement pour calculer la covariance de l'erreur d'estimation et pour gérer les

redistributions. Toutes ces données sont envoyées à la tâche maître pour calculer l'estimé et controler la dégénérescence du support.

A.2 Calcul de l'estimé et de la covariance de l'erreur d'estimation

A partir de ces données $(Z_k^j, X_k^j, \widetilde{P}_k^j, Q_k^j)$ envoyées par les tâches esclaves, la tâche maître T_0 calcule le coefficient de normalisation des poids :

$$S_k = \sum_{j=1}^{P} Z_k^j \tag{16}$$

l'estimé $\widehat{x}_k$ (9) et la covariance $\widetilde{P}_k$ de l'erreur d'estimation (10) :

$$\widehat{x}_k = \frac{\sum_{j=1}^{P} X_k^j}{S_k} \qquad \text{et} \qquad \widetilde{P}_k = \frac{\sum_{j=1}^{P} \widetilde{P}_k^j}{S_k} - \widehat{x}_k^T \widehat{x}_k \tag{17}$$

A.3 Redistribution et normalisation

Pour vérifier la dégénérescence du support global, la tâche maître T_0 calcule le nombre de particules efficaces (13) :

$$N_{eff} \approx \frac{(S_k)^2}{\sum_{j=1}^{P} Q_k^j} \tag{18}$$

Si le nombre de particules efficaces est en dessous du seuil préalablement fixé, la redistribution globale du support particulaire est nécessaire. Dans ce cas, la tâche maître affecte une variable booléenne b_k à 1 et renvoie à l'ensemble des tâches esclaves $T_{j=1...P}$ le coefficient de normalisation des poids S_k ainsi que l'indicateur de dégénérescence b_k.

A la réception de ce message, les tâches esclaves vérifient la nécessité de redistribution en examinant la variable booléenne b_k. Si aucune redistribution n'est nécessaire ($b_k = 0$), elle procèdent à la prochaine itération $k+1$ de la boucle principale. Dans le cas contraire, la procédure de redistribution décrite dans la section suivante a lieu.

B. Redistribution et équilibre de charge

Si une redistribution est nécessaire ($b_k = 1$), les tâches esclaves $T_{j=1...P}$ normalisent leurs poids $\rho_k^{i,j}$:

$$\rho_k^{i,j} \longleftarrow \frac{\rho_k^{i,j}}{S_k} \tag{19}$$

et les supports particulaires locaux $\left\{x_k^{i,j}, \rho_k^{i,j}\right\}_{i=1,...,n_j^k}$ sont redistribués localement proportionellement à leur poids II-B. Aucune communication inter-tâches n'est nécessaire.

Le problème de cette approche est que le nombre de particules n_k^j affectées à chacun des processeurs n'est plus équilibré après redistribution. Asymptotiquement un seul processeur risque de capitaliser la totalité de la charge de calcul. Pour équilibrer la charge de calcul après redistribution, il suffit d'échanger les particules entre tâches esclaves. La procédure d'équilibrage de charge est illustrée sur la figure 3.

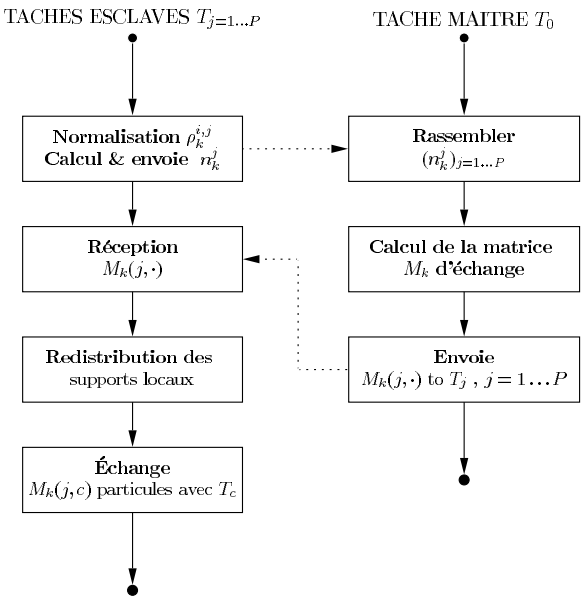


Fig. 3. Procedure d'équilibrage des charges

Le nombre de particules à échanger entre chaque couple de tâches esclaves est calculé par la tâche maître. A cette fin, les tâches esclaves $T_{j=1...P}$ calculent et envoient a la tâche maître la taille de leur support après redistribution, i.e :

$$n_k^j = \sum_{i=1}^{n_{k-1}^j} PE\left[\rho_k^{i,j} N\right] \tag{20}$$

Ces données sont rassemblées par la tâche maître qui peut ainsi calculer le nombre total de particules obtenu après redistribution $N_r = \sum_{j=1}^{P} n_k^j$, ainsi que l'écart Δn_k^j pour chacune des tâches esclaves $T_{j=1...P}$ par rapport à la charge idéale $PE(N_r/P)$:

$$\Delta n_k^j = n_k^j - PE(N_r/P) \tag{21}$$

Ces écarts sont utilisés pour construire une matrice d'échange M_k assurant l'équilibre de charge. $M_k(l,c)$ désigne le nombre de particules échangées entre la tâche T_l et la tâche T_c. Si $M_k(l,c) > 0$, le flot de particules va de T_l vers T_c. Évidemment $M_k(l,c) = -M_k(c,l)$ car le nombre de particules envoyées et reçues sont identiques. Cette matrice M_k est calculée selon l'algorithme suivant :

Algorithm 1 (Calcul de la matrice d'échange).

Debut

Pour $l = 1 \ldots P$ **faire**
  Si $(\Delta n_k^l > 0)$ /* T_l *envoyer* */
  Pour $c \neq l$ **faire**

Si $(\Delta n_k^c < 0)$ /* T_c *reçoit de* T_l */
Si $(\Delta n_k^l + \Delta n_k^c \leq 0)$ /* T_l *envoie tout à* T_c */
$M_k(l,c) = \Delta n_k^l$
$M_k(c,l) = -\Delta n_k^l$
$\Delta n_k^c = \Delta n_k^c + \Delta n_k^l$
$\Delta n_k^l = 0$
Autrement /* T_c *reçoit tout de* T_l */
$M_k(l,c) = -\Delta n_k^c$
$M_k(c,l) = \Delta n_k^c$
$\Delta n_k^l = \Delta n_k^l + \Delta n_k^c$
$\Delta n_k^c = 0$
Fin SiAutrement
Fin Si
Fin Pour
Fin Si
Fin Pour
Fin.

La tâche maître envoie ensuite aux tâches esclave $T_{j=1,...,P}$ la $j^{\text{ème}}$ ligne de M_k notée $M_k(j,\cdot)$ qui leur correspond.

A la réception de $M_k(j,\cdot)$, les tâches esclaves $T_{j=1,...,P}$ effectuent leur redistribution locale puis des échanges selon l'algorithme suivant :

Algorithm 2 (Politique d'échange des tâches esclaves $T_{j=1,...,P}$).

Début
Pour $c = 1 \ldots P$ **and** $c \neq j$ **faire**
Si $M_k(j,c) > 0$
Envoyer $M_k(j,c)$ particules de T_c
Autrement Si $M_k(j,c) < 0$
Recevoir $M_k(j,c)$ particules à T_c
Fin SiAutrement
Fin Pour
Fin.

Après échange, le nombre de particules résultant de la tâche $T_{j=1...P}$ est :

$$n_k^j = \sum_{i=1}^{n_{k-1}^j} round\left[\rho_k^{i,j} N\right] + \sum_{c=1}^{P} M_k(c,j) \qquad (22)$$

Cela conduit à un équilibrage des charges de calcul quasi optimal. Bien entendu, cette procédure est assez coûteuse en termes de communications. Mais, étant donné que la redistribution est peu fréquente, son coût est négligeable en regard du bénéfice apporté par l'équilibrage de charge.

Une amélioration de cette procédure, qui minimise les coûts de communication, consiste à intervertir les procédures d'échanges et de redistribution. A la place d'echanger un grand nombre de particules de poids proches de $1/N$, les particules échangent un nombre réduit de particules de poids plus importants.

IV. Analyse de performances

L'algorithme parallèle développé avec la bibliothèque de communication MPI a été testé sur le problème de trajectographie passive SONAR pour cibles manoeuvrantes (voir [3]). Ce problème difficile et fortement non-linéaire fournit un "benchmark" convaincant. Trois scénarios, de granularité croissante, i.e. à nombres de particules croissants, sont étudiés :

	$SCNA$	$SCNB$	$SCNC$
N	500	5000	100 000

Les résultats présentés ont été obtenus sur deux types d'architectures parallèles :

- **SGI/Cray Origin2000.** L'Origin2000 est une architecture multi-processeur (32-PE CC-NUMA) . Les processeurs sont des 64-bit R10000 (250 MHz), et les noeuds sont interconnectés par des commutateurs CrayLinks (800 MB/s full duplex).
- **Grappe de PC.** Il s'agit d'un réseau de 5 biprocesseurs Intel-PII-233 (10 PEs) interconnectés par de Ethernet 100Mbps. L'objectif est d'étudier les performances de ce type d'architecture peu onéreuse sur une application concrète du FP.

Les performances obtenues sont moyennées sur 50 trajectoires de bruits d'observation distinctes. Les tables I et II indiquent la durée par itérations itération en fonction du nombre de processeurs sur chacune des architectures. La performance de l'Origin 2000 est, comme cela était prévisible, bien supérieure à celle du cluster de PC (de l'ordre de 60 %). Cependant, un simple cluster de PC assure le temps réel (de l'ordre de $5s$) sur le scénario C avec seulement 4 taches esclaves.

TABLE I
Temps moyen par itération (ms) sur Origin2000

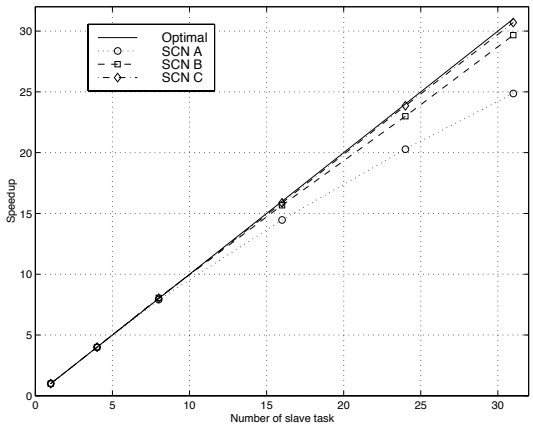
Number of PE	4	8	16	24	32
SCN A	18.8	9.4	5.2	3.7	3.0
SCN B	186.5	92.8	47.5	32.5	25.1
SCN C	3763	1877	949	631	485

TABLE II
Temps moyen par itération (ms) sur cluster de PC

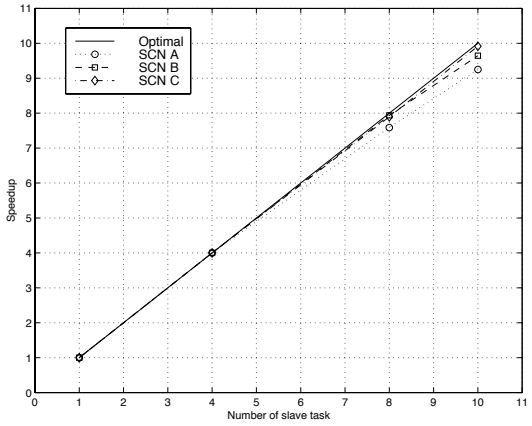
Number of PE	4	8	10
SCN A	29.6	15.6	12.8
SCN B	301.7	152.0	125.1
SCN C	5660	2867	2283

Les accélérations (speedup) obtenues en fonction du nombre de taches esclaves sont représentées sur la figure 4. L'Origin2000 présente une accélération quasi linéaire jusqu'à 32 PE (l'efficacité avec 32 PE est supérieure à 80% pour le scénario A). La courbe d'accélération pour le scénario A s'incurve plus rapidement sur le cluster de PC que sur l'Origin 2000. Ce phénomène est principalement du aux fabiles performances du réseau Ethernet.

Deux conclusions s'imposent à la vue de ces résultats :



Origin 2000



PC cluster

Fig. 4. Accélération moyenne en fonction du nombre de taches esclaves

• La problème de trajectographie SONAR pour cibles manoeuvrantes peut être traité en temps réel sur une architecture parallèle peu onèreuse.
• Pour le scénario A, de granularité très faible, avec 32 PE, seulement 15 particules sont assignées à chacun des processeurs. Or l'efficacité reste supèrieure à 80 %. La courbe d'accélération croît certainement pour un nombre de PE bien plus important. Toutefois, l'accélération ne peut croître indéfiniment. Si le grain devient trop petit, d'autres versions parallèles doivent être envisagées pour minimiser les communications entre taches esclaves et maîtres.

– La matrice d'échange pourrait être calculée "a priori" par la tache maître, en effet :

$$round\left[\frac{Z_k^j}{S_k}Nc\right]$$

forunit une bonne approximation du nombre de particules redistribuées. Ainsi, une seule communication serait nécessaire pour équilibrer la charge. La charge ne serait pas parfaitement équilibrée. Mais, le gain en terme communications serait très significatif pout un grand nombre de PE.

– L'estimé peut être delivré périodiquement toutes les p itérations. Cela diviserait le coût de communication par p. Toutefois, cette période doit être choisie judicieusement car la redistribution ne pourrait avoir lieu entre deux périodes successives.

V. Conclusion

Nous avons montré dans cet article que la structure parallèle inhérente au FP peut être exploitée très efficacement par une approche directe.

L'algorithme développé a une très grande scalabilité et les performances obtenues sont compatibles avec le temps réel pour le problème de trajectographie passive SONAR pour cibles manoeuvrantes sur des architectures peu onéreuses comme une grappe de PC (i.e 2 ou 3 biprocesseurs). Il est clair que des résultats similaires sont prévisibles sur d'autres applications encore plus complexes étant donné que cette complexité ne peut qu'entraîner un grain plus gros.

Nous développons actuellement de nouvelles approches pour étendre la scalabilité de l'algorithme à des centaines de processeurs. Le but étant de traiter un grand nombres de particules ($> 50K$), sur des problèmes où le rapport signal sur bruit est très faible et où la cadence d'échantillonnage est élevée comme dans le cas du signal RADAR.

References

[1] J.S LIU and R. CHEN *Blind deconvolution via sequential imputations* Journal of the American Statistical Association, Vol. 430, 1995.
[2] O. BRUN, J. M. GARCIA *Real-Time Parallel Particle Filtering* Fourteenth International Symposium of Mathematical Theory of Networks and Systems, Perpignan, France, 2000.
[3] V. TEULIERE *Contribution au filtrage de Volterra et à l'estimation particulaire - Application aux transmissions électroniques et acoustiques* PhD Thesis, Université Paul Sabatier - LAAS/CNRS, 2000.
[4] H. CARVALHO and P. DEL MORAL and A. MONIN and G. SALUT *Optimal nonlinear filtering in GPS/INS integration* IEEE Transactions on Aerospace and Electronic Systems, Vol. 33, No 3, 1997.
[5] N.J GORDON and D.J SALMOND and A.F.M SMITH *Novel approach to nonlinear/non-Gaussian bayesian state estimation* IEE-Proceedings-F, Vol. 140, No 2, 1993.
[6] J.C. NOYER *Traitement non-linéaire du signal radar par filtrage particulaire* PhD Thesis, Université Paul Sabatier - LAAS/CNRS, 1996.
[7] T. HUILLET and G. SALUT *Interprétation des équations du filtrage non-linéaire* Séance du GdR Automatique du CNRS (Pôle non linéaire), 1989.
[8] J-C NOYER, G. SALUT *Optimal non-linear radar tracking of non-Gaussian manoeuvering targets* to appear in IEEE Aerospace and Electronic Systems
[9] A. MONIN, G. RIGAL, P. CHARRON, *A New Technology for LORAN-C Receivers* International Symposium on Integration of LORAN-C/Eurofix and EGNOS/Galileo, Bonn, Germany, 22-23 march 2000.
[10] JAZWINSKI. Stochastic processes filtering theory. Academic Press, 1970.