UNIVERSITÉ DE
SHERBROOKE
Faculté de génie
Génie électrique et génie informatique

Implémentation d'un algorithme de localisation, suivi et séparation de sources sonores sur DSP pour un robot mobile

Mémoire de maîtrise ès sciences appliquées
Spécialité : génie électrique

Simon BRIÈRE

Sherbrooke (Québec) Canada 08 2007

Your file Votre référence
ISBN: 978-0-494-37843-4
Our file Notre référence
ISBN: 978-0-494-37843-4

RÉSUMÉ

Un robot mobile est un agent qui doit interagir avec plusieurs éléments de son environnement. Les objets et les personnes sont quelques-uns des éléments dont le robot mobile doit se soucier. Un des modes d'interactions les plus intuitifs pour un humain est la parole. Cependant, l'interaction avec les sons n'est pas une tâche aisée pour un robot, et c'est pour cette raison que le système de localisation, suivi et séparation de sources mobiles AUDIBLE a été créé. Ce système rend possible la localisation et le suivi simultané de quatre sources sonores, ainsi que la séparation de trois sources sonores.

Bien souvent, les robots mobiles se doivent d'être compacts et d'avoir une bonne autonomie énergétique. Or, le système actuel nécessite l'usage d'un ordinateur portable qui doit être installé sur le robot et qui consomme une bonne quantité de puissance électrique. Afin de rendre le système plus portable et efficace dans un contexte d'installation sur des systèmes embarqués, il devient nécessaire de diminuer ces contraintes en réduisant la taille du système, la puissance de calcul nécessaire de même que ses contraintes énergétiques.

Afin de réaliser cette implémentation, un processeur de type DSP (*Digital Signal Processor*) point flottant a été utilisé. En raison de la complexité de l'algorithme, plusieurs ajustements et compromis ont dû être réalisés pour transférer le système sur ce processeur. Néanmoins, le système embarqué parvient à localiser et suivre deux sources sonores à la fois, et à séparer ces deux sources. La localisation sur le système embarqué obtient une précision moyenne de 3.1° pour l'azimut et de 3.5° pour l'angle d'élévation, comparativement à une précision de 1.1° (azimut) et 0.89° (élévation) pour le système original. Le système permet une précision qui est aussi grande sinon plus que celle de l'oreille humaine. Le suivi donne également des résultats comparables au système original, malgré les simplifications qui ont été apportées.La séparation, quant à elle, est plus limitée en raison de contraintes temporelles. Lorsqu'au moins une source est séparée, la durée maximale des échantillons sonores qui peuvent être séparés par le système varie entre 5.2 (pour deux sources suivies) et 64 secondes (pour une source suivie), rendant le système plus efficace avec des sources de courte durée. Les taux

de reconnaissance vocale des sources séparées sont comparables entre les deux systèmes, obtenant un résultat de 89.6% pour le système embarqué et de 92.8% pour le système original avec des échantillons de voix humaine de courte durée.

Ce travail démontre donc qu'il est possible de porter le système AUDIBLE sur des systèmes embarqués ayant des limitations plus importantes en terme de puissance de calcul. Le système est un bon point de départ pour rendre plus accessible les capacités auditives sur des robots de tout genre.

REMERCIEMENTS

Le premier remerciement va à M. Jean-Marc Valin, qui est l'auteur original du système d'audition AUDIBLE. Sans sa patience, ses réponses rapides aux questions malgré sa distance et son implication dans le projet, il n'aurait pas été possible de mener celui-ci à terme.

Un remerciement particulier doit être fait à mon directeur de recherche, M. François Michaud, pour son apport académique et technique tout au long du projet.

Il faut également remercier tous les membres du groupe de recherche LABORIUS qui ont été impliqués de près ou de loin dans ce projet. Parmi ces gens, il faut citer M. Dominic Létourneau qui a été sollicité par de nombreuses questions, M. Serge Caron pour la conception du cube de test des microphones et M. Daniel Labonté pour ses réflexions sur le choix du microprocesseur.

Il ne faut pas oublier non plus la collaboration de M. Yan Morin de RoboMotio inc. pour la conception des cartes de microphones.

Finalement, un merci particulier à ceux qui ont été impliqués dans les tests et qui ont soit fourni une aide logistique ou qui ont généreusement fait don de leur voix pour les tests de séparation. Merci également à Stéphanie Gauthier pour son aide dans les tests de suivi de sources et pour son support moral.

Une partie de ce projet a été financé par les Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT).

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

CHAPITRE 1

INTRODUCTION

Afin de permettre à un robot mobile d'être efficace dans son environnement, il faut le doter de capacités sensorielles adéquates. Il existe présentement plusieurs senseurs et capteurs permettant à un robot de reconnaître les éléments qui composent son environnement. En effet, un robot peut, par exemple, utiliser la vision artificielle (à l'aide de caméras) pour reconnaître des couleurs ou lire des affiches [19], ou encore les capteurs de proximité (sonars, infrarouges, laser) pour lui permettre de naviguer parmi des obstacles. De la même façon, pour interagir avec des gens, le robot doit être doté de capteurs et de composantes facilitant les échanges. Pour ce faire, un robot doit être capable de parler mais aussi d'entendre et d'écouter.

Plusieurs recherches sont présentement en cours sur le sujet. Il s'agit cependant d'un domaine de recherche plutôt jeune. En effet, les premiers efforts sur le domaine de l'audition artificielle appliqué à la robotique mobile ont débuté en 1995 [17]. Le but principal du domaine est de parvenir à recréer un système d'audition similaire à celui possédé par les êtres humains. En ce moment, la façon d'y parvenir consiste à utiliser une série de microphones disposés de façon à capter les sons originant de plusieurs directions. Par la suite, un traitement doit être effectué sur les signaux reçus de façon à séparer les différentes sources sonores, et ce même si ces sources sont entremêlées (ce qui survient par exemple lorsqu'un groupe de personnes discutent ensemble, phénomène couramment référé dans la littérature comme étant le *cocktail party effect*).

Le projet actuel repose sur un système développé au LABORIUS (Laboratoire de Robotique Mobile et Systèmes Intelligents de l'Université de Sherbrooke). Le système consiste à traiter les signaux provenant de huit microphones afin de localiser, suivre et séparer des sources sonores entremêlées [38]. Le système fonctionne présentement sur un ordinateur dédié installé sur un robot mobile. Cependant, comme il s'agit d'un système préparant les signaux reçus

à un traitement ultérieur (e.g., la reconnaissance de parole), l'intégration de celui-ci directement dans l'ordinateur n'est pas la solution optimale. En effet, le système nécessite une bonne puissance de calcul. Or, l'ordinateur dédié doit gérer plusieurs éléments, et ne peut consacrer une trop grande partie de son temps sur ce système. C'est pourquoi ce projet se concentre sur la modularisation du système en le transférant sur un DSP (*Digital Signal Processor*). En transférant le système sur un processeur indépendant, on décharge l'ordinateur principal et il n'est plus nécessaire d'installer un ordinateur portable dédié pour effectuer le traitement.

D'autres contraintes motivent également le transfert vers un système embarqué. L'espace s'avère souvent une contrainte en robotique mobile, surtout lorsque la taille des robots tend à diminuer. Il est donc souhaitable d'avoir un système de plus petite taille, portable et modulaire. La durée de l'autonomie énergétique est également un critère important : on veut maximiser le temps que le robot pourra fonctionner avec ses piles internes. Finalement, un autre objectif à long terme de la robotique est de diminuer le coût des plate-formes robotiques. Pour ce faire, un système embarqué devrait s'avérer moins coûteux qu'un ordinateur complet.

Le présent travail est rédigé sous la forme d'un mémoire par article et est divisé en quatre sections. Le chapitre 2 présente une revue des systèmes qui utilisent des systèmes d'audition, des implémentations précédentes sur des processeurs embarqués et des robots qui sont influencés par les sons de leur environnement. Le chapitre 3 présente des informations supplémentaires sur l'implémentation réalisée, les choix de conception et les compromis effectués. Le chapitre 4 présente un article portant sur la méthodologie et l'implémentation du système auditif sur le DSP. Finalement, le chapitre 5 présente des informations complémentaires sur les tests présentés dans l'article et sur des tests supplémentaires qui ont été effectués sur le sous-système de séparation.

CHAPITRE 2

AUDITION ET ROBOTIQUE

Avant de se lancer dans le vif du sujet, il convient de présenter une revue des travaux passés ou en cours sur des sujets similaires. Ainsi, le cadre théorique du projet de recherche se divise en quatre principaux éléments : l'usage de divers systèmes de microphones dans divers contextes auditifs, l'algorithme principal à la base du projet, les implémentations précédentes de systèmes parents sur DSP et finalement les plate-formes robotiques utilisant un système d'audition et la façon dont leur système est implémenté et utilisé.

2.1 Systèmes de microphones

Pour permettre à un robot d'entendre, il faut l'équiper d'un ou de plusieurs microphones. L'usage d'un seul microphone comporte plusieurs limitations : qualité sonore limitée, impossibilité de localiser et de traquer adéquatement des sources sonores, incapacité à dissocier et séparer des sources sonores entremêlées. Il faut donc utiliser plus d'un seul microphone pour des applications nécessitant de telles fonctionnalités. L'usage d'une série de microphones disposés dans l'espace (communément appelé *microphone arrays*) font l'objet de recherches dans plusieurs domaines.

Tout d'abord, dans le domaine de la téléconférence, l'usage d'un tel système s'avère utile pour capter correctement la voix des personnes impliquées et ce, peu importe l'acoustique de la pièce dans laquelle ils se trouvent. Le système présenté par Kellermann [18] utilise trois séries de 11 microphones. Chaque série se spécialise dans un domaine de fréquence : basses, moyennes et hautes. Le système utilise également le principe du *Beamforming* (voir section 2.2.1) et un système de votation pour sélectionner le microphone qui capte la meilleure source sonore. Le système a été implémenté sur un ordinateur dont les spécifications ne sont pas précisées. Le traitement s'effectue en temps réel, à un taux d'échantillonnage de 8 kHz. Le

temps de réaction du système était suffisant pour ne pas constater de délais, mais aucune valeur quantitative n'a été présentée pour confirmer ce fait.

Ensuite, un système de microphones peut être utilisé pour faciliter la reconnaissance vocale, comme le font Giuliani et al. [14]. En effet, un tel système améliore la réception des signaux à traiter, puisque l'usage de plusieurs microphones (quatre dans ce cas-ci) permet de couvrir plus d'une dimension spatiale. De plus, le traitement effectué sur les signaux reçus permet d'améliorer la qualité pour faciliter la reconnaissance. Un logiciel de reconnaissance de voix (APASCI) est par la suite utilisé pour reconnaître ce qui a été dit. Un système similaire peut également être utilisé pour permettre l'identification d'une personne à partir de sa voix [20]. La localisation de la personne qui parle dans l'environnement s'effectue à l'aide d'un *Beamformer*. Le traitement de ces informations est effectué en temps réel par un ordinateur (dont les spécifications ne sont pas précisées).

Un système composé de plusieurs microphones permet également la séparation de plusieurs sources sonores entremêlées (référé généralement comme étant le *cocktail party effect* [40]). Le nombre de microphones utilisés dans ce genre de système varie selon le système. Deux microphones sont utilisés par Rennie et al. [32] afin d'évaluer un système probabiliste pour la séparation d'un nombre de sources variant entre deux et quatre.

L'usage de plusieurs microphones rend également possible la localisation d'une ou plusieurs sources sonores dans l'environnement. Par exemple, Gatica et al. [13] utilisent un système comportant huit microphones et une caméra vidéo dans le but de détecter la position d'une personne qui parle. Également, Aarabi et al. [1] utilisent un véhicule qui se localise grâce à 24 microphones fixés sur les murs de la pièce. Le système arrive à localiser la position du véhicule en utilisant une mesure de *Time Delay Of Arrival* (TDOA, voir section 2.2.1), ce qui permet d'indiquer au véhicule émettant un son constant où il se trouve dans la pièce. Le système permet également au véhicule de localiser avec précision une personne qui parle dans la pièce.

Finalement, des systèmes composés de microphones sont également utilisés afin de fabriquer des prothèses auditives pour les personnes qui ont des difficultés d'audition. Un tel système

utilise une certaine quantité de microphones (neuf dans le cas de Chowdhury [9]) de très petite taille et qui sont montés sur une prothèse. L'avantage d'utiliser ici plusieurs microphones au lieu d'un seul réside dans le filtrage du bruit et permet donc d'améliorer la qualité du son perçue par l'utilisateur de la prothèse.

2.2 Algorithme à la base du travail

Le présent travail repose entièrement sur le système développé par Jean-Marc Valin [36]. Seul un aperçu du système est présenté dans cette section. Plus de détails sont disponibles dans le chapitre 4 ainsi que dans les articles de référence. Le système a été développé sur le robot Spartacus que l'on peut voir à la figure 2.1. Les huit microphones composant le système se trouvent aux endroits identifiés par une "*" sur la figure.

La figure 2.2 illustre les différents éléments qui composent l'algorithme. À l'entrée du système, on retrouve plusieurs sources sonores qui peuvent prendre diverses formes : clappement de main, parole, bruit de sirène, etc. Ces sources sonores sont entremêlées entre elles et le bruit ambiant, lui, peut provenir de diverses sources : système de ventilation, bruit des moteurs du robot, etc. Le son environnant est capturé par une série de microphones, huit dans le montage original. Le nombre de microphones du système peut varier entre quatre et huit. Cependant, par les tests effectués, il a été déterminé que le système ne pouvait localiser correctement plus de deux sources en utilisant uniquement quatre microphones [36]. Des microphones omnidirectionnels peu dispendieux sont utilisés avec de très bons résultats. Étant donné les variations matérielles entre chacun des microphones, le gain de ceux-ci est ajusté de façon logicielle afin de s'assurer qu'ils possèdent tous la même réponse à un signal donné.

Le système est composé principalement de trois sous-systèmes : le premier servant à la localisation de sources sonores, le second au suivi des sources sonores localisées et le troisième à la séparation des sources sonores localisées. Le but du système est de parvenir à séparer les sources sonores entre elles et de limiter l'impact du bruit sur la qualité de ces sources. En plus de les séparer, le système permet de localiser spatialement les sources dans l'espace. L'algorithme a été testé avec un nombre de sources sonores variant de une à quatre [38].

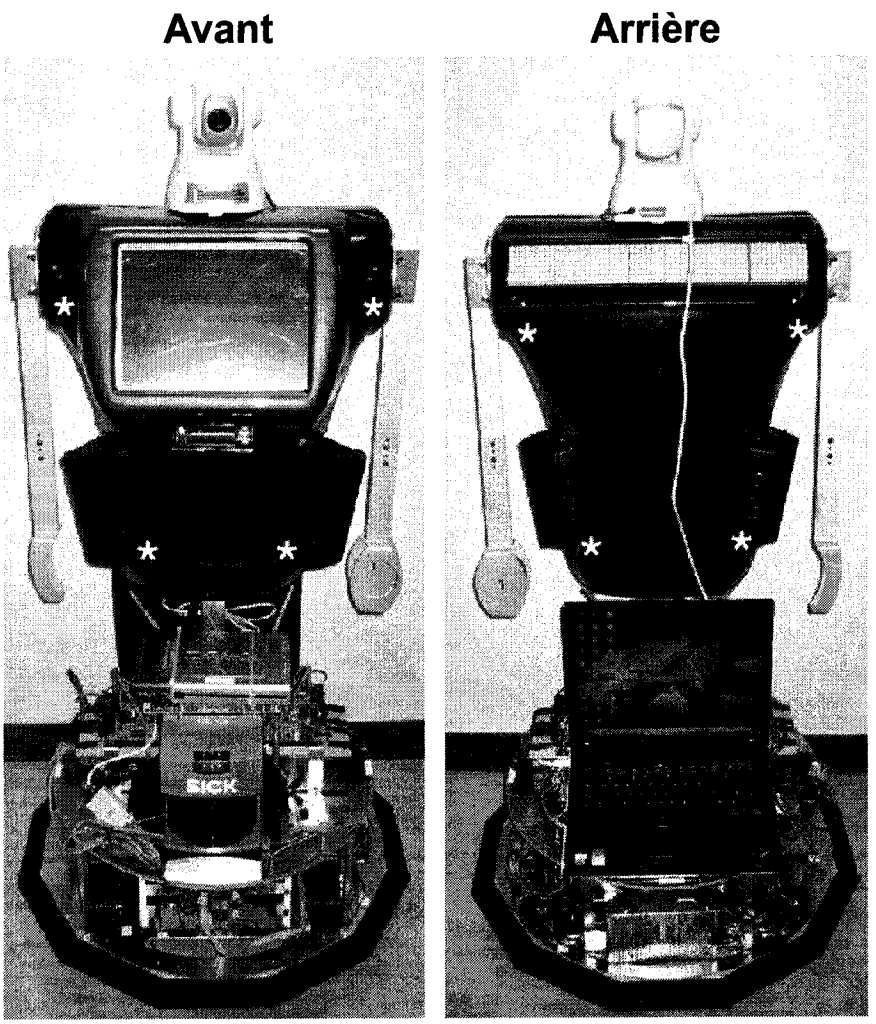

Figure 2.1 – Le robot Spartacus ayant servi au développement d'AUDIBLE

2.2.1 Localisation de source(s) sonore(s)

Une fois les signaux sonores capturés par les microphones, il faut localiser ces sources sonores dans l'espace environnant. Pour ce faire, il existe deux principales approches. Ces approches sont énumérées à la figure 2.2 dans le sous-système de localisation.

La première de ces approches est l'approche par *Time Delay of Arrival* (TDOA) [21]. L'approche consiste à prendre les délais d'arrivée du son à chacun des microphones. Pour parvenir à quantifier ce délai, il faut établir une mesure de cohérence. La mesure de cohérence la plus commune est la corrélation croisée simple entre les signaux perçus par deux microphones. Cette mesure peut être exprimée comme suit :

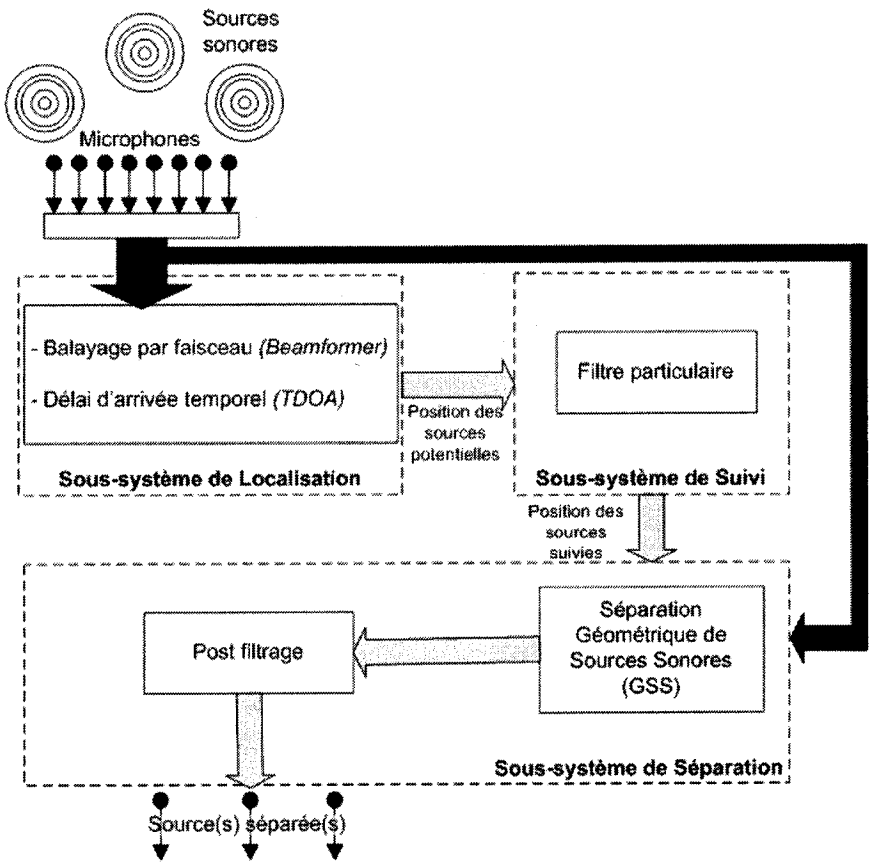

Figure 2.2 Schéma global de l'algorithme de base du projet

$$R_{ij}(\tau) = \sum_{n=0}^{N-1} x_i[n]x_j[n-\tau] \tag{2.1}$$

où $x_i[n]$ est le signal reçu par le microphone i, $x_j[n]$ est le signal reçu par le microphone j, N est le nombre d'échantillons par trame et τ est le décalage de la corrélation en échantillons.

Un des premiers problèmes avec l'équation 2.1 réside dans la complexité de l'algorithme, soit un algorithme de l'ordre de $O(N^2)$. Pour réduire la complexité de l'algorithme, on peut calculer une approximation dans le domaine fréquentielle, réduisant la complexité à $O(Nlog_2N)$.

Le problème majeur de cette approximation réside dans le fait que la corrélation dépend beaucoup des propriétés statistiques des signaux. Étant donné que les signaux qui sont d'intérêt pour le système (voix) sont généralement passe-bas, la corrélation entre deux échantillons adjacents est grande et génère des pics de corrélation croisée qui peuvent être très larges.

Pour résoudre ce problème, il faut introduire un bruit blanc avant d'effectuer la corrélation croisée. La méthode présentée par Omologo [27] est utilisée. On obtient alors l'équation 2.2, qui correspond à la transformée de Fourier inverse du spectre croisé (*cross-spectrum*) normalisé :

$$R_{ij}(\tau) \approx \sum_{k=0}^{N-1} \frac{X_i(k)X_j(k)^*}{|X_i(k)|\,|X_j(k)|} e^{j2\pi k\tau/N} \tag{2.2}$$

où $X_i(k)$ est la transformée de Fourier discrète de $x_i(n)_t)$, $X_i(k)X_j(k)^*$ est le spectre croisé de $x_i(n)_t)$, $x_j(n)_t)$, $(.)^*$ dénotant le conjugué complexe et $e^{j2\pi k\tau/N}$ correspondant au bruit introduit.

Malheureusement, l'introduction de ce bruit apporte un autre problème. Chacune des plages fréquentielles contribue du même facteur au calcul de la corrélation finale, et ce même si la plage est dominée par le bruit, rendant le système moins robuste au bruit. En augmentant le poids des plages qui contiennent une rapport signal sur bruit (SNR) élevé, on parvient à contrer ce problème et à obtenir l'équation 2.3 :

$$R_{ij}(\tau) = \sum_{k=0}^{L-1} \frac{\zeta_i(k)X_i(k)\zeta_j(k)X_j(k)^*}{|X_i(k)|\,|X_j(k)|} e^{j2\pi k\tau/L} \tag{2.3}$$

où $\zeta_i(k)$ est le facteur de pondération pour la fréquence k qui prend en considération à la fois le bruit et la réverbération en utilisant une technique de MCRA (*Minima-Controlled Recursive Average*) [10].

Par la suite, ayant cette mesure de corrélation croisée, il est possible de calculer le TDOA ΔT_{ij} entre le microphone i et j en trouvant la plus grande valeur dans la corrélation croisée :

$$\Delta T_{ij} = argmax_\tau R_{ij}^{(e)}(\tau) \tag{2.4}$$

En utilisant N microphones, il est possible de calculer $N(N-1)/2$ corrélations croisées différentes. En utilisant par la suite des calculs géométriques, il est possible de trouver la

position de la source sonore. Il est à noter qu'en tant que telle, cette façon de faire n'est applicable que s'il n'y a qu'une seule source sonore à détecter, quoique la détection de plus d'une source est possible en théorie. Cette approche fonctionne avec une précision de 3° dans un rayon de 3 mètres [39]. L'environnement utilisé pour qualifier la précision de la localisation était un environnement avec un niveau de bruit assez élevé. Aucun test n'a cependant été réalisé dans des environnements moins bruités.

La deuxième approche utilisable dans le système est une approche qui utilise la technique du *Beamformer* [38]. C'est cette approche qui est utilisée dans le système final. La technique consiste en un premier temps à séparer l'environnement en plusieurs points d'intérêt et à balayer chacun de ces points pour y détecter la présence de sources sonores. Les points sont générés en créant une grille icosahédrale (20 points). Chacun des triangles composant la surface est alors sous-divisé en quatre sections triangulaires égales. Ce processus est répété quatre fois, ce qui nous donne 2562 points disposés sur une surface unitaire. Chacun de ces points couvre un rayon d'environ 2.5° autour de son centre. La figure 2.3 illustre la grille de recherche avec 2562 points.

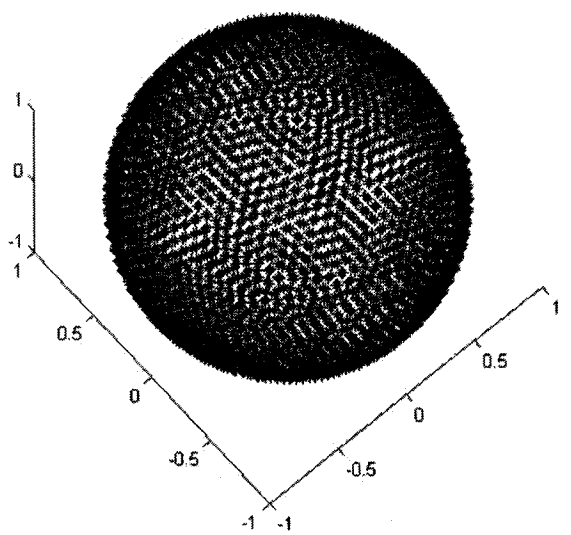

Figure 2.3 – Sphère utilisée pour la recherche de sources sonores

Par la suite, chacun des points formant la grille de recherche est balayé par le faisceau. Le but est de déterminer la direction dans laquelle l'énergie du faisceau est la plus grande. La sortie d'un *beamformer* de type *delay-and-sum* est définie par l'équation 2.5 :

$$y(n_t) = \sum_{n=0}^{N-1} x_n(n_t - \tau_n) \tag{2.5}$$

où $x_n(n_t)$ est le signal du n^{ieme} microphone et τ_n est le délai d'arrivée pour ce microphone. L'énergie de sortie du *beamformer* sur une trame de L échantillons peut être définie par [36] :

$$E = K + 2 \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{n_1-1} R_{x_{n_1}, x_{n_2}}(\tau_{n_1} - \tau_{n_2}) \tag{2.6}$$

où K est pratiquement constant et peut être ignoré lorsque l'énergie est maximisée. Le calcul de l'énergie est effectué sur tous les points de la grille en effectuant des corrélations croisées sur les TDOA de chacune des paires de microphones et par rapport à la direction du point de la grille considéré. Une source est considérée présente dans la direction où l'énergie est la plus grande. Les temps d'arrivée du son sont calculés de la façon suivante, en utilisant l'hypothèse que les sources ne sont pas situées à proximité des microphones (*far-field assumption*) :

$$\tau_{ij} = \frac{F_s}{c}(\vec{\mathbf{p}_i} - \vec{\mathbf{p}_j}) \cdot \vec{\mathbf{u}} \tag{2.7}$$

où F_s est la fréquence d'échantillonnage de la carte d'acquisition, c est la vitesse du son, $\vec{\mathbf{p}_i}$ et $\vec{\mathbf{p}_j}$ sont les positions respectives des microphones i et j et $\vec{\mathbf{u}}$ est un vecteur unitaire pointant dans la direction du point.

Cette technique permet de localiser plus d'une source. L'algorithme doit alors avoir un nombre fixe de sources S à trouver. Le balayage s'effectue alors S fois, les corrélations croisées des sources déjà détectées étant ignorées.

Étant donné que la grille offre une précision de 2.5° dans chaque direction, une étape optionnelle peut être effectuée pour raffiner la détection en effectuant une estimation de la distance à laquelle la source se trouve. Pour ce faire, une grille de 125 points est définie autour du point où l'énergie du faisceau était la plus grande. Cette grille donne une erreur maximale d'environ 1°. Étant la façon dont la grille est générée, l'hypothèse que les sources ne sont plus situées à proximité ne tient plus et l'équation 2.7 doit être redéfinie de la façon suivante :

$$\tau_{ij} = \frac{F_s}{c}(\left\|d\,\vec{\mathbf{u}} - \vec{\mathbf{p}_j}\right\| - \left\|d\,\vec{\mathbf{u}} - \vec{\mathbf{p}_i}\right\|) \tag{2.8}$$

où d est la distance entre la source et le centre de l'arrangement des microphones. L'évaluation de la distance d se fait pour des valeurs entre 50 cm et 5 m. Cependant, il a été constaté que la valeur de d n'était pas assez fiable pour fournir une bonne estimation de la distance des sources [36] et qu'en conséquence, cette méthode ne peut être utilisée pour donner une distance précise de la source localisée.

2.2.2 Suivi de sources sonores par filtre particulaire

Les approches utilisées dans le sous-système de localisation de sources sonores donnent des données valides à un temps t. Or, il n'est pas directement possible de suivre ces sources, étant donné que le robot peut se déplacer et que les sources peuvent également se déplacer. De plus, le sous-système de localisation trouve un nombre de sources fixe (quatre dans le système original) peu importe qu'il y ait présence de quatre sources ou non. Le filtre particulaire utilisé dans le sous-système de suivi permet de solutionner ces problèmes.

Il a été démontré qu'un filtre particulaire pouvait s'avérer efficace pour suivre une source [2]. Le filtre particulaire fonctionne en modélisant par un point chacune des positions potentielles d'une source sonore dans l'espace. Chacun de ces points est pondéré par un poids directement proportionnel à la certitude que possède le système face à la présence d'une source sonore à cet endroit, et est caractérisé par son état (arrêté, source en accélération, source à vitesse constante), sa position et sa vitesse. Le filtre permet de confirmer la présence d'une source sonore (à partir de probabilités et de la sortie du système de localisation) et d'estimer la trajectoire de celle-ci. Il permet également de gérer l'apparition et la disparition de sources sonores.

L'implémentation du filtrage particulaire est réalisé avec le modèle de prédiction présenté par Ward et al. [42] qui permet de prédire la position et la vitesse de chacune des particules.

Le filtre doit ensuite, utilisant les probabilités, classifier les sources détectées par le sous-système de localisation dans une des trois catégories suivantes :

- Fausse détection ($H0$) ;
- Source actuellement suivie ($H1$) ;
- Nouvelle source qui n'est pas actuellement suivie ($H2$).

Le poids de chacune des particules des filtres est ensuite mis à jour avec la probabilité que la source soit une vraie source, et les sources pour lesquelles le niveau de certitude est inférieur à 0.3 (valeur fixée empiriquement) sont supprimées des sources envoyées au sous-système de séparation, alors que les sources dont la probabilité est supérieure à 0.98 sont ajoutées à la liste des sources utilisées pour la séparation. Une fois les poids mis à jour, les filtres sont ré-échantillonnés si de nouvelles données sont disponibles pour une source donnée afin d'éviter que les particules ne convergent toutes vers la même position (i.e. une particule possédant un très grand poids, alors que les autres tendent vers un poids égal à 0). Le ré-échantillonnage est effectué en utilisant une technique de ré-échantillonnage basée sur l'importance de chacun des échantillons (*Sampling Importance Resampling*) est utilisée. Cette technique consiste à générer un nouvel ensemble de particules en utilisant un modèle probabiliste qui accorde aux particules ayant les poids les plus élevés une plus grande importance [2]. Le ré-échantillonage a également comme effet d'uniformiser le poids de chacune des particules du filtre.

2.2.3 Séparation de source(s) sonore(s)

Une fois les sources sonores localisées et suivies par le sous-système de suivi de sources sonores, il est possible de les séparer et de filtrer le bruit environnant. Pour ce faire, une approche par séparation géométrique de sources (*Geometric Source Separation — GSS*) [28] est utilisée.

En posant $S_m(k,l)$ comme étant la source sonore m à la trame l et la fréquence discrète k, nous obtenons l'équation 2.9, qui correspond au signal reçu par les microphones :

$$x(k,l) = \mathbf{A}(k)s(k,l) + n(k,l) \tag{2.9}$$

où $s(k,l)$ est le vecteur correspondant à la source $S_m(k,l)$, $n(k,l)$ est le bruit ambient non-cohérent reçu aux microphones, et $\mathbf{A}(k)$ est une matrice pouvant être estimée avec les

résultats du sous-système de localisation et de suivi de sources. Le résultat de la séparation est donné par l'équation 2.10 :

$$y(k,l) = \mathbf{W}(k,l)x(k,l) \tag{2.10}$$

où $\mathbf{W}(k,l)$ est la matrice de séparation. Cette matrice doit être estimée en posant les deux contraintes suivantes :

- Les sorties séparées doivent être décorrelées ;
- Le gain doit être unitaire dans la direction de la source d'intérêt et nul dans les directions des interférences.

Une fonction de coût est créée pour chacune des contraintes, et une phase d'adaptation permet d'ajuster la matrice de séparation.

Au niveau de la première contrainte, le système GSS original nécessite plusieurs secondes de données pour estimer les matrices de corrélation. Pour pallier à cette contrainte qui nuit au traitement en temps réel des données, une approche par gradient stochastique est utilisée. Cette approche utilise des estimations instantanées et permet à l'algorithme GSS de fonctionner en temps réel.

Occasionnellement, l'algorithme GSS peut occasionner l'apparition de grandes valeurs dans la phase d'adaptation. Il est souhaitable d'avoir des valeurs le plus petit possible dans la matrice de séparation et un terme de régularisation est alors utilisé pour diminuer ces valeurs.

L'approche GSS simplifiée permet de séparer adéquatement trois sources sonores avec un rapport signal sur bruit (SNR) entre 10 dB et 14 dB [40].

2.2.4 Post filtrage

Une fois les sources séparées, il est nécessaire de les passer dans un filtre afin de soustraire le bruit ambiant et l'interférence présents dans chacune des sources. Le post filtrage considère que toutes les interférences (à l'exception du bruit ambiant) proviennent de sources sonores détectées. L'estimation de la variance du bruit $\lambda_m(k,l)$ est déterminée par l'équation 2.11.

$$\lambda_m(k,l) = \lambda_m^{stat.}(k,l) + \lambda_m^{leak}(k,l) + \sum_{i=0}^{M-1} \lambda_i^{rev}(k,l) \tag{2.11}$$

où $\lambda_m^{stat.}(k,l)$ est l'estimation du bruit ambiant calculé avec la technique du MCRA [10], $\lambda_m^{leak}(k,l)$ est l'estimation de la fuite (*leak*) entre les sources (pouvant être causé par des erreurs dans la localisation des sources, différences dans les réponses fréquentielles des microphones, effets de champ rapproché, etc.) et $\lambda_i^{rev}(k,l)$ est l'estimation de la réverbération. Pour calculer l'estimation de fuite, on assume que l'interférence des autres sources est réduite d'un facteur η (variant entre -10dB et -5dB) en raison de l'algorithme de séparation. L'estimation est donc calculée de la façon suivante :

$$\lambda_m^{leak}(k,l) = \eta \sum_{i=0, i\neq m}^{M-1} Z_i(k,l) \tag{2.12}$$

où $Z_m(k,l)$ est le spectre lissé de la source m $Y_m(k,l)$, défini de façon récursive comme étant :

$$Z_m(k,l) = \alpha_s Z_m(k,l-1) + (1-\alpha_s)\,|Y_m(k,l)|^2 \tag{2.13}$$

où α_s=0.2.

Dans un deuxième temps, le post filtrage ajuste le gain des sources sonores dans le but d'en faire le traitement avec un système de reconnaissance vocale. Le gain est calculé en prenant comme hypothèse que de la parole est présente dans la trame considérée. Le gain optimal est calculé selon l'équation 2.14.

$$G(k) = \{G_{H_1}(k)\}^{p(k)} \cdot {G_{min}}^{1-p(k)} \tag{2.14}$$

où $G_{H_1}(k)$ est le gain spectral optimal défini par Ephraim et Malah [11], $p(k)$ est la probabilité de présence de la parole et G_{min} correspond au gain minimum alloué lorsque la parole est absente (-20 dB).

La probabilité de la présence de la parole est estimée selon l'équation 2.15.

$$p(k) = \left\{1 + \frac{\widehat{q}}{1-\widehat{q}}(1+\xi(k))exp(-\upsilon(k))\right\}^{-1} \tag{2.15}$$

où $\widehat{q}$ correspond à la probabilité *a priori* de l'absence de parole telle que définie par Cohen et Berdugo [10], $\xi(k)$ est le rapport signal sur bruit (SNR) *a priori* défini par Ephraim et Malah [11] et $\upsilon(k) = \gamma(k)\xi(k)/(\xi(k)+1)$ où $\gamma(k)$ est le rapport signal sur bruit (SNR) *a posteriori*, également défini par Ephraim et Malah [11].

2.3 Utilisation de DSP dans des systèmes similaires

Présentement, il existe très peu de projet utilisant des DSP (*Digital Signal Processor*) pour réaliser un système de localisation et de séparation de sources sonores dans le domaine de la robotique mobile.

Stoytchev utilise un système de localisation de sources sonores stéréo (à deux microphones) sur un robot mobile [34]. Son système est implémenté sur un DSP qui permet de donner la direction générale d'une source sonore de même que son intensité. Cependant, il est important de mentionner que le focus de cette recherche se concentre surtout sur le développement d'une architecture décisionnelle robotique et non pas sur un système de traitement audio.

Un système se rapprochant sans doute davantage du projet est le système développé par Rabinkin [30]. Le système a été conçu dans le but d'être utilisable pour faire de la téléconférence, afin de permettre de se focaliser directement sur les gens qui parlent sans avoir besoin d'un opérateur dédié à la tâche. Le système en question utilise une série de huit microphones qui sont branchés sur une carte DSP. La carte en question n'est cependant pas autonome ; elle est reliée à un ordinateur via une interface de type PCI. Le DSP sert uniquement à effectuer l'acquisition des données et aucun calcul n'y est effectué. Les données recueillies par la carte sont par la suite traitées par l'ordinateur. Le DSP fonctionne à 30 MHz, et l'acquisition audio est effectuée à 16 kHz sur 16 bits. Le processeur est un DSP à point flottant. Les résultats se sont avérés concluants, l'algorithme permettant de positionner correctement une source

sonore avec un taux d'erreur d'environ 0.5% dans une pièce avec peu de réverbération (3 m par 4 m). Par contre, dans une pièce de 10 m par 15 m avec beaucoup de réverbération, le système s'est avéré moins efficace, avec un taux d'erreur d'environ 17%.

Inspirée des travaux de Rabinkin, une expérimentation a par la suite été effectuée par Silverman [33]. Dans son expérimentation, Silverman a utilisé 128 DSP similaires à ceux utilisés par Rabinkin pour faire l'acquisition et le traitement de 512 microphones. Le système était également utilisé pour localiser des sources sonores situées dans l'environnement. Des FPGAs (*Field Programmable Gate Array*) ont été utilisés pour gérer le bus de communication. Le système s'est avéré efficace, puisqu'un temps de traitement (localisation et acquisition) de 150 ms a été atteint, l'objectif étant d'avoir un temps inférieur à 500 ms.

2.4 Des robots qui entendent

Il y a quelques robots qui ont été développés et qui comportent un système d'audition. Ces robots utilisent des microphones à diverses fins : reconnaissance de parole, localisation de sons, navigation, etc.

Une des méthodes les plus simples pour capturer une source sonore est d'utiliser un microphone qui est positionné le plus près possible de la source audio. De cette façon, il n'est pas nécessaire de se préoccuper de la séparation des sources sonores ni même de leur localisation dans l'espace. C'est ce système qui est utilisé par Kismet, le robot émotif [3]. En effet, Kismet capture la voix des gens qui désirent lui parler grâce à un microphone sans fil qui est positionné près de la bouche. La figure 2.4 montre une interaction avec Kismet.

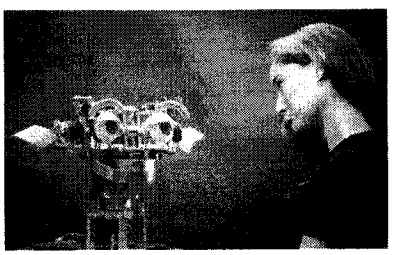

Figure 2.4 – Le robot Kismet (MIT), tiré de [3]

La plupart des robots capables d'entendre utilisent un système composé de deux micro-

phones. Ce système a comme but de s'apparenter au système d'audition humain qui possède deux capteurs (oreilles). Cependant, un tel système est limité puisqu'il ne permet pas de distinguer si ce qui est entendu est situé à l'avant ou à l'arrière de l'axe passant par les deux microphones.

Parmi les robots utilisant deux microphones, il y a tout d'abord le robot utilisé par Stoytchev [34] dans le but de développer son architecture robotique. Celui-ci utilise deux microphones et une carte DSP dans le but de localiser la direction générale d'une source sonore et son intensité. Dans l'architecture présentée, le robot utilise ses capteurs sonores pour déterminer si quelqu'un désire lui parler (via un clapement de mains).

Il y a également le robot BIRON [15], montré à la figure 2.5. Ce robot utilise un système composé de deux microphones dans le but de localiser les sources sonores se trouvant à l'avant du robot. Le robot utilise son système d'audition afin d'établir une communication avec des personnes. Il combine le son qu'il reçoit avec ce qu'il voit afin de déterminer si une personne lui adresse la parole ou non. Son système d'audition lui permet également d'effectuer une reconnaissance de la parole. Le traitement du système auditif est effectué avec l'ordinateur embarqué sur le robot (Pentium III, 850MHz) et le traitement des données s'effectue en temps réel. Le système, combiné avec le système de capture et de traitement vidéo permet de suivre adéquatement une personne [12].

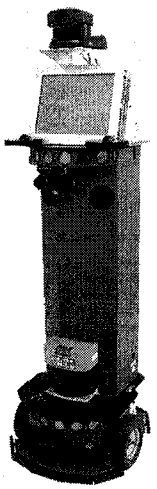

Figure 2.5 – Le robot BIRON (Bielefeld University), tiré de [15]

Le robot DAV (figure 2.6), développé à l'Université du Michigan, utilise également un système

de microphones [16]. Ce robot est l'évolution du robot SAIL [44]. Le robot possède deux microphones installés de chaque côté de la tête du robot. Les microphones sont utilisés afin de faire de la reconnaissance vocale. Les instructions vocales sont interprétées et permettent au robot d'apprendre de nouvelles actions. Une fois entraîné, le robot est capable de reconnaître les instructions vocales en temps réel, à l'aide de son ordinateur embarqué (dont les spécifications exactes ne sont pas précisées). Lorsqu'entraîné avec 15 commandes répétées 5 fois par 11 personnes différentes, le robot parvient à reconnaître et à identifier la commande dans presque 100% des cas. Aucune mention n'est faite des performances du robot lorsque qu'il y a plusieurs sources sonores d'intérêt (i.e. plusieurs locuteurs) dans l'environnement qui parlent en même temps ,ni de l'effet de la position de la personne par rapport au robot lors des tests.

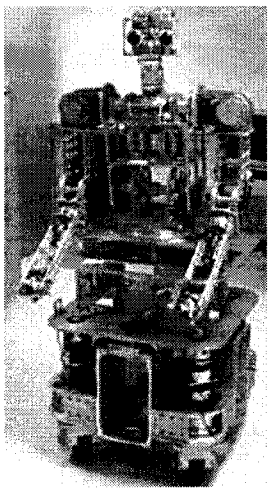

Figure 2.6 – Le robot DAV (Université du Michigan), tiré de [16]

Un autre projet, TeleHead II, utilise deux microphones dans le but de localiser une source sonore [35]. Ce projet est développé dans le but de réaliser un robot opéré à distance. La conception et la réalisation de ce projet repose sur l'hypothèse que l'usage d'un système d'audition faciliterait le contrôle à distance. La localisation des sources sonores se fait de façon différentielle entre les deux microphones. Le robot localise ainsi la direction de la source sonore selon l'intensité reçue à chacun des microphones.

Le robot ROBITA, quant à lui, utilise ses deux microphones pour suivre une conversation entre plusieurs personnes [22]. Les microphones sont utilisés afin d'orienter le visage du

robot vers la personne qui parle lors de la conversation. Un système de localisation monté en différentiel, similaire à celui du robot DAV, est utilisé. La figure 2.7 illustre le robot.

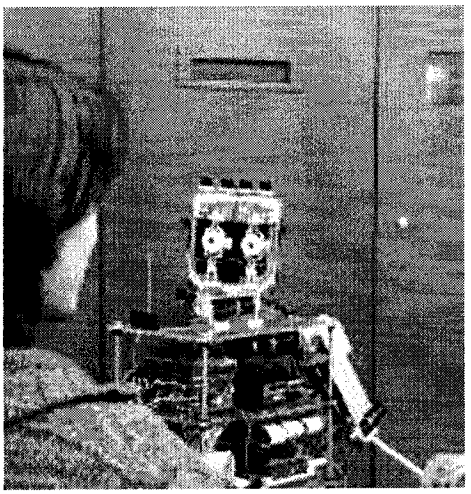

Figure 2.7 – Le robot ROBITA (Université du Wasada), tiré de [22]

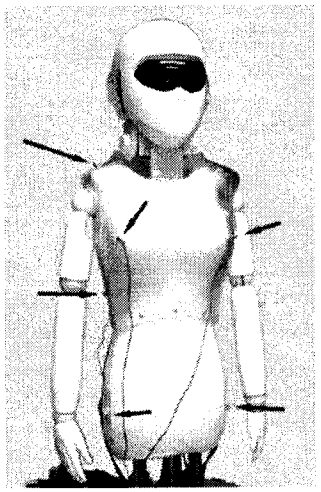

Figure 2.8 – Le robot SIG2 (Université de Kyoto), tiré de [43]

Un des seuls robots à utiliser plus de deux microphones est le robot SIG2 [43]. SIG, son prédécesseur, utilisait deux microphones tandis que SIG2, montré à la figure 2.8, utilise huit microphones afin d'utiliser le système de séparation de Valin [36]. Le robot est utilisé pour faire de la reconnaissance vocale et le système de séparation des sources sonores décrit dans la section 2.2.3 est utilisé. Le traitement est effectué en temps réel, sur un ordinateur de type Pentium M 1.6 GHz. Une fois les sources séparées, elles sont envoyées à un système de reconnaissance de la parole. Tout dépendemment de l'angle d'incidence de la source sonore

par rapport au robot, les taux de succès de la reconnaissance varient entre 15% (pour une source située à 60° à gauche du robot) et 78.5% (pour une source située à l'avant du robot).

Huit microphones sont également utilisés dans la réalisation d'un robot de service [8]. Ces microphones sont disposés tout autour du robot de façon à former un cercle. Ces microphones sont utilisés dans le but d'effectuer une reconnaissance de la parole.

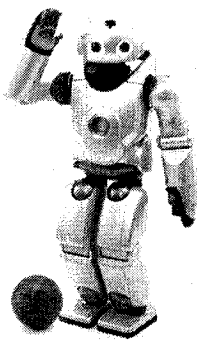

Figure 2.9 – Le robot QRIO (Sony), tiré du site officiel du QRIO (fermé fin mars 2007)

Un robot possédant des capacités d'interaction audio peut également s'avérer utile dans un contexte de stand d'information. C'est dans ce but que Brock utilise un robot de type B21R afin d'en faire un stand d'information mobile [5]. Le robot utilisé est représenté à la figure 2.10. Ce robot combine un écran plat, un système utilisant quatre microphones pour faire la localisation de la parole et des bruits ambiants, et d'une caméra stéréo combinée avec un laser pour effectuer la détection de visages. Le robot donne de l'information sur un sujet donné, et augmente le volume si le bruit ambiant devient trop élevé. De même, il change d'endroit s'il est incapable de parler plus fort que ce bruit.

Au niveau commercial, Sony a emboîté le pas en réalisant un robot qui utilise sept microphones. Ce robot, le QRIO, utilise les microphones situés dans sa tête pour reconnaître des mots et pour identifier des personnes de façon à pouvoir les reconnaître par la suite. Les techniques utilisées pour permettre la mise en oeuvre de ce système ne sont cependant pas dévoilées. QRIO est représenté à la figure 2.9. Le QRIO a cependant cessé d'exister en janvier 2006.

Enfin, au niveau militaire, l'armée américaine expérimente avec un robot muni de huit mi-

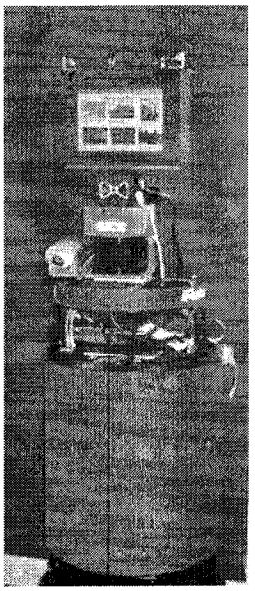

Figure 2.10 – Le robot B21R d'iRobot, utilisé comme stand d'information mobile, tiré de [5]

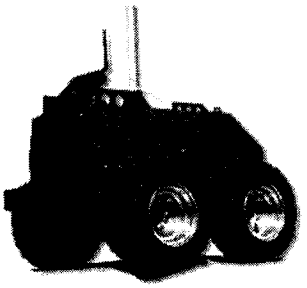

Figure 2.11 – Le robot ATRV-2 (iRobot)

crophones montés sur un cercle de 16 pouces (environ 38 cm) afin de localiser des coups de feu. Le tout a été monté sur un robot commercial, le ATRV-2 de iRobot (figure 2.11). Un système utilisant une approche basée sur le TDOA (voir section 2.2.1) est utilisée, mais en utilisant trois microphones à la fois au lieu de deux. Le traitement est effectué en temps réel sur un petit ordinateur portatif embarqué (Toshiba Libretto). Le système s'avère efficace (en localisant la position d'un coup de feu dans 95% des cas), mais son efficacité est réduite de beaucoup dans un environnement où il y a beaucoup d'écho ou dans un environnement où plusieurs coups de feu sont tirés dans un laps de temps très court. Afin de remédier à ces problèmes, les chercheurs travaillent à faire une version utilisant 16 microphones.

2.5 En résumé

Des systèmes de microphones sont utilisés dans diverses applications : téléconférence, prothèses auditives, reconnaissance vocale, localisation spatiale et séparation de plusieurs sources sonores entremêlées. C'est dans le but de résoudre les deux dernières problèmatiques que le système AUDIBLE utilisé sur un robot mobile a été développé. En plus de permettre la localisation et la séparation de plusieurs sources sonores, il permet également le suivi de celles-ci dans les cas où les sources se déplaceraient ou que le robot serait en mouvement. Le but du présent travail est de rendre ce système plus pratique dans un contexte de robotique mobile en le transférant sur un processeur de type DSP. Quelques systèmes auditifs ont déjà été implémentés sur DSP, mais aucun ne permet la localisation, le suivi et la séparation de plusieurs sources sonores simultanément. Il existe cependant plusieurs robots qui bénéficient et exploitent des capacités auditives dans le but, généralement, de localiser des sources sonores et d'effectuer une reconnaissance vocale sur les trames recueillies. Le présent travail est donc justifié, puisque le besoin de capacités auditives en robotique mobile est présent, et que ce besoin doit être comblé en répondant aux contraintes importantes de la robotique, soit la portabilité du système, le traitement en temps réel et la faible consommation énergétique.

CHAPITRE 3

PLATE-FORME DE DÉVELOPPEMENT ET CONCEPTION

Considérant le fait que le nombre de pages dans l'article écrit comme élément central du présent ouvrage était limité, les informations qui y sont présentées n'entrent pas dans tous les détails du travail réalisé. Il convient donc de préciser dans ce chapitre certains des éléments qui sont une part important du projet, soit l'implémentation qui a été effectuée sur la plate-forme de développement et les choix de conception qui ont dû être faits afin d'arriver à porter et à adapter AUDIBLE, le système d'audition réalisé par Valin, sur une plate-forme à microprocesseur embarqué.

3.1 Choix de la plate-forme

Un des éléments clés du projet consiste à choisir une plate-forme adéquate pour supporter l'implémentation de l'algorithme. En mettant de côté les éléments matériels complémentaires (i.e., USB et convertisseurs analogiques - digitaux), le type de processeur choisi est l'élément central le plus important.

Il apparaît évident qu'on peut déjà éliminer les processeurs embarqués non dédiés au traitement de signaux et plus adaptés aux systèmes de contrôle (microcontrôleurs). En effet, en raison de la quantité de calculs devant être effectués par la carte, de tels processeurs n'offrent pas de performance suffisante.

Les processeurs de type FPGA (*Field Programmable Gate Array*) sont excellents pour effectuer des calculs en parallèles très rapidement. Dans le système de localisation, suivi et séparation de sources, les calculs sont effectués sur des vecteurs et donc sont hautement parallèles. Cependant, la taille du système est un facteur important. Les FPGAs se caractérisent

par un nombre de portes maximales pouvant être utilisées. Les FPGAs hautes performances possèdent jusqu'à 6 millions de portes, mais leur coût est très élevé, sans compter les coûts reliés à la carte qui doit supporter le FPGA. Des FPGA comportant entre 40000 et 50000 portes sont disponibles, mais leur coût demeure encore élevé par rapport à des processeurs plus conventionnels. Pour avoir un coût comparable, il faut considérer des FPGAs ayant entre 8000 et 20000 portes logiques. Il est difficile d'estimer le nombre de portes requises par AUDIBLE, mais ce nombre risque d'être assez élevé en raison de la taille du système et de la quantité de données qu'il doit traiter. L'usage d'un FPGA unique a été mise de côté pour des raisons de coût et de l'incertitude reliée à la taille requise.

Le choix s'est donc tourné vers les processeurs dédiés au traitement de signal, les DSPs. Ces processeurs apportent une puissance de calcul suffisante pour le traitement de signal à grande échelle. Ces processeurs sont classés en deux grandes familles : les processeurs à point fixe et les processeurs à point flottant. La différence se situe au niveau de l'architecture interne du processeur qui est optimisée pour un type de représentation numérique en particulier. Chaque famille comporte des avantages et des inconvénients, dont certains sont donnés dans le tableau 3.1.

TABLEAU 3.1 – Comparaison entre DSP à point fixe et à point flottant

Type de DSP	**Avantages**	**Inconvénients**
Point flottant	- Conversion du code actuel simplifiée - Troncation des nombres non nécessaire	- Moins de librairies de fonctions disponibles présentement - Consommation électrique plus élevée - Coûts généralement plus élevés
Point fixe	- Beaucoup de librairies de fonctions disponibles - Convient mieux à une commercialisation en raison de la disponibilité	- Troncation des nombres à point flottant - Temps de développement plus élevé en raison des optimisations supplémentaires requises

Il a donc été décidé de prendre un DSP à point flottant pour ce projet. Cette décision a été justifiée par la simplification de la conversion du code actuel, de même que par les

cartes commerciales déjà disponibles pour interfacer huit entrées audio. En effet, le code actuel utilise déjà largement des opérations en point flottant. Les problèmes (e.g., perte de précision) reliés à la conversion des opérations de type point flottant en opérations de type point fixe sont évités.

Une solution de type "hybride" où un processeur DSP serait jumelé à un processeur de type FPGA a également été rejetée, principalement en raison des coûts reliés à l'ajout d'un FPGA et au temps de développement qui aurait été augmenté.

3.2 Architecture matérielle

La figure 3.1 représente les différents composants du système. Au cœur du système, on retrouve le processeur DSP TMS3206713 sur lequel sont branchés les différents composants. La version du processeur qui est utilisée fonctionne avec une horloge de 225 MHz. Il s'agit d'un processeur DSP à point flottant qui possède 256 kilo-octets de mémoire interne et qui permet la configuration de caches de type L1 et L2 dans celle-ci. Selon les spécifications du fabricant, le processeur est capable d'exécuter 1800 MIPS (*Million Instructions Per Second*) et 1350 MFLOPS (*Million Floating point Operations Per Second*). La carte sur laquelle le DSP est installé possède 64 Mbytes de mémoire externe.

En utilisant l'interface EMIF (*External Memory Interface*) qui est intégrée au processeur, la mémoire externe (SDRAM) peut être adressée en utilisant le bus de données ED et le bus d'adresses EA. Le bus de données fonctionne à 100 MHz. Un étage de tampon (*buffer*) est utilisé afin d'interfacer le contrôleur USB. En raison de l'architecture d'accès mémoire utilisé ici (défini par la carte de développement qui a été utilisée), l'interface USB passe par les mêmes bus que l'interface mémoire, ce qui a pour effet de ne pas permettre de transferts USB en même temps qu'un accès mémoire.

Pour effectuer la capture des échantillons audio, le bus McASP0 (*Multichannel Audio Serial Port 0*) est utilisé. Ce bus spécifique à la famille 6000 des DSP points flottants de TI (*Texas Instruments*) permet le transfert direct des données du bus vers la mémoire interne du DSP à un taux pouvant aller jusqu'à 192 kHz. Les quatre convertisseurs analogues/digitaux

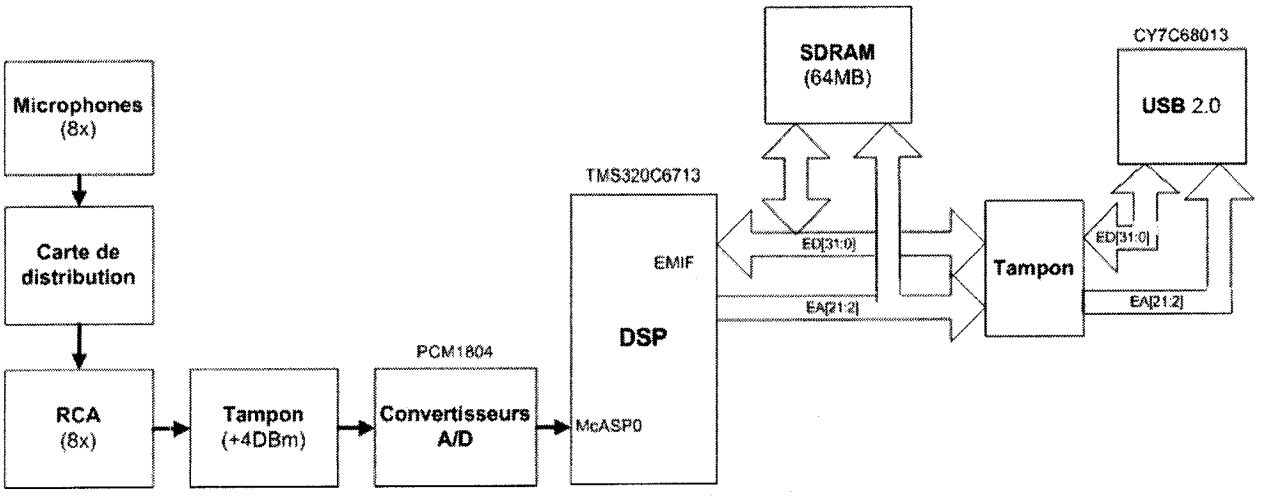

Figure 3.1 – Schéma bloc des composantes matérielles du DSP

sont branchés sur le bus en parallèle, et un contrôleur permet de synchroniser ceux-ci. Les convertisseurs fonctionnent à 32 kHz, avec 24 bits de précision et chacun d'entre eux est en charge de convertir deux canaux audio, soit deux microphones. Un étage de tampon fournit un gain de 4 dBm aux signaux d'entrée, qui correspondent à huit connecteurs de type RCA. Le signal des huit microphones est branché à ces connecteurs via une carte de distribution, qui agit comme point central pour distribuer les signaux entre les connecteurs RCA (utilisés par la carte DSP) et un connecteur de type DB9 (utilisé par la carte de capture du système original). Chaque microphone est également monté sur une carte permettant d'ajuster le gain pour compenser les variations rencontrées entre chacun d'entre eux.

La carte de développement utilisée offre une multitude d'autres interfaces vers d'autres connecteurs et formats, mais ceux-ci ne sont pas utilisés dans l'implémentation actuelle du système et donc ne sont pas décrits ici.

Au niveau de la consommation énergétique, la carte de développement est fournie avec un bloc d'alimentation de 30 W. Un ordinateur portable équivalent à celui utilisé par le système AUDIBLE original (Pentium M 1.6 GHz) fonctionne avec un bloc d'alimentation d'environ 90 W. Évidemment, il est difficile de quantifier la consommation précise de la carte de développement et de l'ordinateur portable puisque les différentes composantes (carte d'acquisition, stockage, mémoire, etc.) présentes dans chacun des systèmes et l'usage qui en sont fait viennent influencer la consommation. En se concentrant uniquement sur les

processeurs, selon les spécifications des fabricants, le DSP TMS3206713 consomme environ 1 W lorsque tous ses sous-systèmes(EMIF, interruptions, etc.) sont utilisés à 50% et que l'unité de calcul fonctionne à 60%. Du côté de l'ordinateur portable, la consommation énergétique varie de 5 W (lorsque le processeur est au repos) à 27 W (lorsqu'il fonctionne à pleine capacité). Il est donc évident avec ces nombres que l'usage d'un système embarqué réduit de beaucoup la consommation énergétique requise par le système.

3.3 Architecture logicielle

La structure de l'implémentation réalisée sur le DSP est relativement semblable à celle utilisée pour l'implémentation originale. Quelques choix ont dû cependant être faits, et la structure a dû être modifiée pour l'adapter à la configuration du DSP. Les sections suivantes apportent des précisions supplémentaires sur la conception logicielle qui n'ont pas été couvertes par l'article.

3.3.1 Flux de données

Le flux de données est un flux séquentiel, à l'exception des interruptions. Comme indiqué à la figure 3.2, la première étape est l'initialisation des différents systèmes. Les initialisations qui sont effectuées lors de cette étape n'ont pas été optimisées pour le processeur, puisqu'elles ne sont effectuées qu'une seule fois et que leur temps d'exécution n'est pas critique dans les performances du système.

Dans un premier temps, les composantes matérielles (la carte en tant que telle, les convertisseurs A/D, le transfert mémoire A/D, le port USB et les interruptions) sont initialisées. Dans un deuxième temps, les paramètres logiciels sont générés. Ces paramètres consistent principalement en la génération des points constituant la sphère de 2562 points utilisée pour la localisation, du calcul des délai d'arrivée des différents sons à chacune des paires de microphone (TDOA) en fonction de chaque point de la sphère, à la génération des tableaux utilisés pour effectuer les FFT et à l'initialisation des différents sous-systèmes, soit la localisation, le suivi et la séparation. De plus, les interruptions sont paramétrées et activées.

Deux interruptions différentes sont utilisées dans le système. La première interruption est appelée lorsqu'un transfert de données via EDMA (*Enhanced Direct Memory Access*) est complété pour une trame de 1024 échantillons pour les huit microphones. Cette interruption copie les données nouvellement transférées vers le tampon de trames, et active le drapeau indiquant qu'il y a des données en attente. La seconde interruption est appelée lorsque des données USB doivent être envoyées. Cette interruption pourrait également être utilisée pour recevoir des données USB (par exemple, des commandes de contrôle), mais cette fonctionnalité n'est pas implémentée dans la version du système actuelle.

Une fois les initialisations complétées, le système démarre. La boucle principale attend que des données soient en file dans le tampon de trames. Lorsqu'au moins une trame est présente, on exécute tout d'abord le sous-système de localisation. Ce sous-système est lui-même divisé en deux sections. La première section met à jour les paramètres d'estimation de bruit et de l'énergie du signal qui sont requis par le *beamformer*. Cette section est exécutée à toutes les itérations du système. La deuxième section, quant à elle, utilise le *beamformer* pour parcourir les 2562 points de la sphère de localisation et ainsi trouver deux sources potentielles. Comme le système recherche toujours deux sources, il y a un fort taux de fausses détection à cette étape. Néanmoins, le tout est détecté et corrigé par le sous-système de suivi.

Cette section du sous-système de localisation est exécutée à chaque cinq itérations du système. Dans le système original, cette section était exécutée à chaque quatre itérations. Or, dans l'implémentation sur le DSP, il a été rapidement constaté qu'il était pratiquement impossible d'atteindre les mêmes performances que le système original avec les mêmes paramètres. En augmentant à cinq itérations, le temps de traitement moyen sur une seconde se trouve diminué, puisqu'une grande partie des délais dans la localisation provient de cette cinquième itération (voir les résultats dans l'article au chapitre 4).

Le sous-système de suivi de sources sonores est exécuté également une fois à toutes les cinq itérations. Il met à jour les positions des filtres à particules utilisés pour les sources suivies et élimine les fausses détections de l'étape de localisation. À la sortie de ce sous-système, on retrouve les sources qui sont "réelles", i.e., qui ne sont pas de fausses détections et qui doivent être séparées par le sous-système de séparation.

Le sous-système de séparation est appelé à chaque itération, si au moins une source est présentement suivie par le sous-système de suivi. Le système utilise les positions des sources pour séparer les trames sonores et les stocker en mémoire externe. Ensuite, lorsque la séparation est complétée, si des sources ont été séparées et sont suivies, on envoie par USB leur position seulement si cette dernière a été rafraîchie à cette itération, ainsi que les valeurs de chacun des échantillons pour cette trame.

3.3.2 Sections mémoires

La partition de la mémoire est présentée à la figure 4.3. Toutes les données fournies sont valides lorsque le compilateur fonctionne avec les optimisations maximales (-o3) sans informations de *debug* (option "-g" non spécifiée) et en priorisant la vitesse d'exécution sur la taille du code (option "no -ms"), avec les paramètres du système présentés dans l'article (deux sources suivies et séparées, pas de raffinage directionnel, etc.). Cette section s'attarde principalement sur la mémoire interne. Celle-ci étant plus limitée en taille que la mémoire externe, des choix ont dû être faits pour une utilisation efficace. Tout ce qui ne se retrouve pas dans la mémoire interne se retrouve dans la mémoire externe.

Dans la structure de la mémoire interne, les instructions (identifiées par "code" dans la figure 4.3) sont sans doute ce qui nécessite le plus d'espace. Il est difficile de diminuer ce nombre étant donné la complexité du système. Il est cependant évident que des optimisations effectuées au niveau assembleur aurait sans doute un impact (positif ou négatif) sur ce nombre.

La taille de la pile (*stack*) a été ajustée en fonction de la taille maximale requise par le système. Contrairement aux recommandations suggérées pour le développement sur les DSP, la pile n'a pas été fixée la plus petite possible. Ceci est dû au fait qu'une grande quantité de variables locales sont utilisées dans chacune des fonctions. On peut se permettre d'utiliser une grande pile puisqu'on peut identifier le moment où il y a le plus grand nombre d'appels inclusifs de fonctions, soit au moment où les filtres particulaires sont mis à jour.

Dans les constantes et variables qui sont positionnées dans la mémoire interne, on retrouve

les éléments suivants présentés et justifiés dans le tableau 3.2.

Au niveau de l'espace mémoire réservé pour la cache, après nos tests il s'est avéré nécessaire d'utiliser cette cache pour améliorer les performances d'accès à la mémoire externe. La taille de 64 kOctets a été fixées de façon empirique, après avoir validé des temps d'accès très longs avec une cache de 32 ou 48 kOctets. Il pourrait être envisageable de libérer l'espace occupé par la cache avec une gestion mémoire plus serrée, mais il faudrait alors déterminer quelle fonction doit être accélérée au détriment d'une autre.

Un petit espace est réservé pour le tas (*heap*). Le code a été conçu de façon à n'utiliser aucune allocation dynamique de mémoire de façon à gérer la mémoire de façon plus efficace. Cependant, des fonctions d'affichage (*printf*) nécessitent d'avoir un petit tas pour fonctionner.

3.3.3 Gestion des interruptions

Un autre élément important dans l'implémentation du système sur DSP réside dans la gestion des interruptions. Mal gérées, une interruption peut faire tomber le processeur dans un état de latence où aucune instruction n'est plus exécutée et où celui-ci ne répond plus aux commandes. Le nombre d'interruptions présentes dans le système se trouve au nombre de deux : une interruption pour indiquer que le transfert d'une trame en mémoire est complété et une interruption pour indiquer que des données doivent être transmises sur le port USB.

Les librairies fournies avec le DSP contiennent des fonctions qui ne sont pas interruptibles. Or, ces fonctions sont utilisées à plusieurs endroits dans le système de façon à profiter des optimisations qu'elles procurent. De plus, certaines boucles très serrées sont optimisées par le compilateur. Ces boucles serrées ne peuvent être interrompues, rendant donc la gestion des interruptions complexe.

La gestion des interruptions présentement utilisées dans le système est plutôt simpliste. Les interruptions sont désactivées lors de l'exécution de chacun des sous-systèmes, à l'exception de la localisation. Dans ce sous-système, une gestion plus serrée y est effectuée. Les interruptions peuvent donc survenir entre chaque sous-système. Il devient alors important de

s'assurer que le temps d'exécution de ces sous-systèmes ne dépasse pas le temps maximal alloué entre chaque interruption. Dans ce cas-ci, ce temps correspond au temps entre deux trames, soit 32 ms. Comme aucun sous-système ne requiert autant de temps, la gestion faite présentement est fonctionnelle.

3.4 Transfert du système sur DSP

Pour parvenir à effectuer le transfert et avoir un système fonctionnel sur le processeur DSP, plusieurs compromis ont dû être réalisés. Cette section présente les différents compromis effectués ainsi que les raisons qui motivent ceux-ci. Ces compromis sont présentés sommairement dans l'article du chapitre 4.

Tout d'abord, au niveau de la fréquence d'échantillonnage, il a fallu réduire celle-ci de 48 kHz (sur le système original) à 32 kHz (sur le DSP). Ceci est motivé par le fait que le temps maximal de traitement d'une trame de 1024 échantillons capturées à 48 kHz avec un *overlap* de 50% est de 10.66 ms, alors que le temps de traitement de la même trame à 32 kHz est de 16 ms. Le gain en terme de temps donne au système un temps supérieur pour le traitement. Le passage à 32 kHz n'affecte pas la localisation et le suivi de source, mais la qualité des sources séparées se trouve logiquement réduite. Néanmoins, avec les résultats obtenus (voir chapitres 4 et 5), le taux de reconnaissance ne semble pas être affecté par la réduction de la fréquence d'échantillonnage.

Ensuite, le nombre de sources localisées et suivies a été réduit à deux comparativement à quatre dans le système original. Ce compromis a dû être réalisé encore une fois pour réduire le temps de traitement de moitié dans la deuxième partie du système de localisation, et ainsi parvenir à rencontrer des spécifications de traitement en temps réel telles que rencontrées par le système original décrit à la section 2.2. De même, dans le sous-système de localisation, le raffinage directionnel utilisé dans l'algorithme original a été supprimé. En effet, ce raffinage nécessite de générer 125 points autour du point d'intérêt, et d'effectuer un calcul pour chacun de ces points à des distances variant entre 50 cm et 5 m, avec des incréments de 50 cm. L'impact sur le système est une réduction de la précision de la localisation. Un autre impact

est probablement au niveau de la séparation, puisque ce système dépend de la précision de la localisation. Néanmoins, il est difficile de quantifier l'impact sur celle-ci.

Au niveau du nombre de particules utilisées dans les filtres, le système sur DSP utilise 500 particules alors que le système original en utilise 1000. La valeur de 500 a été fixée de façon empirique afin de réduire le temps de calcul requis dans le sous-système de suivi de sources. Cette valeur permet une réduction du temps de traitement, sans trop affecter la qualité du suivi tel que démontré dans le chapitre 4. Une valeur inférieure rendrait le suivi moins précis, alors qu'une valeur supérieure augmenterait le temps de calcul.

Finalement, pour parvenir à traiter les informations en temps réel malgré les contraintes temporelles, il a fallu utiliser un tampon circulaire. Ce dernier permet d'accumuler des trames de 1024 échantillons de 24 bits et de les traiter lorsque le DSP a du temps de disponible. Lorsqu'une trame est capturée par la carte de capture, celle-ci est placée dans le tampon. Lorsque le système est prêt à traiter une trame, il lit la prochaine trame du tampon. Si jamais une trame n'est pas disponible, il attend la prochaine trame. Étant donné que les temps d'exécution de chacun des sous-systèmes n'est pas le même et que ceux-ci ne sont pas tous exécutés à chaque itération, cette technique permet d'accumuler les trames qui ne pourraient être traitées autrement et de les traiter dans les moments où le processeur n'est pas occupé. Concrètement, pour une période de temps donné, le système traite plus d'échantillons que si chaque trame était traitée au fur et à mesure qu'elle est capturée. La taille des trames effective varie donc selon l'état du tampon et le temps de traitement des sous-systèmes actifs à cette itération. Le traitement s'effectue donc sur des super-trames de taille variable, pour une période de temps donné.

Le tampon a été fixé à une taille empirique de 200 trames, ce qui semble être une valeur suffisante. Diminuer la taille de celui-ci a pour effet de rejeter un grand nombre de trames, principalement lorsque le système est occupé à suivre et séparer deux sources. Augmenter la taille retarderait le moment où le système doit rejeter des trames, mais augmenterait aussi le temps de réaction du système. En effet, celui-ci devrait alors traiter toutes les trames précédents celle qui vient d'être capturée, ce qui occasionne un retard sur la sortie. Même avec un tampon de 200 trames un retard est perceptible, et la taille du tampon devrait être

ajustée en fonction des applications. Dans ce cas-ci, la taille a été fixée de façon à ce qu'un message vocal complet de 6.4 secondes puisse être conserver dans le tampon. Afin d'éviter qu'un trop grand retard ne soit accumulé lorsque le système n'a pas de sources à suivre et séparer, si le tampon dépasse un seuil empirique de 25 trames, les trames du tampon sont effacées. Ceci permet au système de détecter de nouvelles sources sans avoir à traiter ce qu'il a déjà accumulé, mais introduit potentiellement la coupure d'une source qui serait apparue dans les trames effacées.

3.5 Montage expérimental

Les figures 3.3 et 3.4 illustrent les différents composants du montage expérimental. À la figure 3.3, on retrouve les huit microphones montés sur le "cube" de test. Les dimensions du "cube" utilisé sont les suivantes : 37 cm × 32 cm × 31 cm. Chacun des microphones est situé à un sommet du "cube". Sur la figure, les microphones sont identifiés par des "*". Le centre du "cube" est pris comme point d'origine pour le système.

Le montage expérimental utilisé est présenté à la figure 3.4. Le cube (A) est placé au centre d'un cercle ayant un rayon d'un mètre. Des marques ont été faites sur le plancher à chaque 22.5°. Deux hauts-parleurs (B) ont été utilisés pour les tests de séparation, et un seul a été utilisé pour les tests de localisation. Dans les tests de suivi, deux personnes marchaient autour du cube sur un cercle de rayon de deux mètres en lisant un texte. Ces hauts-parleurs sont reliés à un amplificateur qui lui est relié à la sortie audio d'un des ordinateurs afin de jouer les enregistrements nécessaires aux tests.

La carte DSP (C) et les deux ordinateurs portables (un servant de client pour la carte DSP (D) et l'autre exécutant le système original (E)) ont été montés sur un chariot facilitant le changement des configurations.

Les tests décrits dans l'article du chapitre 4 présentent sommairement la méthodologie des tests. Néanmoins, certaines précisions s'avèrent nécessaires, principalement au niveau des échantillons sonores utilisés pour chacun des tests.

Tout d'abord, les échantillons sonores utilisés pour les tests de localisation et de détection contiennent principalement trois types de sons : un clappement de mains, une commande vocale ("*Spartacus, come here*") et un échantillon de bruit blanc (100 ms). L'enregistrement est effectué à une fréquence de 48 kHz et un silence d'environ trois secondes est laissé entre chaque type de son. Les sons sont disposés de façon séquentielle dans l'enregistrement, toujours dans le même ordre, soit : clappement, voix et bruit.

Ensuite, au niveau des tests de suivi, les personnes impliquées dans le test devaient marcher à vitesse constante autour d'un cercle de deux mètres de rayon. L'environnement de test était le même environnement que les tests de détection et de localisation. Le texte lu contenait des citations célèbres (comme, par exemple, des citations de Montaigne telles que : "On construit des maisons de fous pour faire croire à ceux qui n'y sont pas enfermés qu'ils ont encore la raison.") et était lu avec un ton et une vitesse constante. Le but n'étant pas de valider la reconnaissance vocale ou la séparation, le contenu du texte n'a pas d'importance.

Finalement, au niveau des tests de séparation présentés dans l'article, l'enregistrement sonore a été généré à partir d'environ 15 voix de personnes différentes provenant de la base de données d'AURORA [29] prononçant des phrases formées de quatre chiffres, à vitesse différente et avec différents accents. Ces voix sont échantillonnées à 8 kHz. L'enregistrement était composé de 25 échantillons de voix féminine et de 25 échantillons de voix masculine pour chacun des deux canaux (haut-parleurs). L'enregistrement a été conçu de façon à ce que lorsqu'une personne commence à parler sur un canal, l'autre commence en même temps. Il est possible qu'un canal se termine avant l'autre en raison du débit vocal qui varie selon les échantillons. L'enregistrement a également été conçu de façon à ce que chacune des combinaisons de type de voix (féminine, masculine) sur les deux canaux soit présentée. Un silence de six secondes est laissé entre chaque échantillon de quatre chiffres.

3.6 En résumé

Afin de transférer le système AUDIBLE sur une plate-forme indépendante et autonome, plusieurs types de processeurs sont offerts : microcontrôleurs, FPGAs, DSP à point fixe et

DSP à point flottant. Ce dernier type de processeur a été retenu en raison des avantages qu'il apportait dans la réalisation du système : développement plus rapide dû au fait qu'aucune conversion vers un système point fixe n'est requis et que des cartes commerciales audio avec ce processeur sont disponibles. Le système a été réalisé sur la carte CPA-II de Lyrtech. Les principaux éléments qui sont abordés sont l'architecture logicielle qui est implémentée, la gestion de la mémoire interne et externe du processeur, qui s'avère un élément critique dans la réalisation du système embarqué, et la gestion des interruptions. Quelques compromis ont également dû être réalisés, principalement au niveau de la fréquence d'échantillonnage du système et au niveau du nombre maximal de sources sonores pouvant être traitées par le système. Un système de tampon circulaire a également été mis en place pour permettre le traitement en temps réel. Finalement, un montage expérimental mettant en parallèle le système original et le système embarqué a été conçu et utilisé dans les tests présentés aux chapitres suivants.

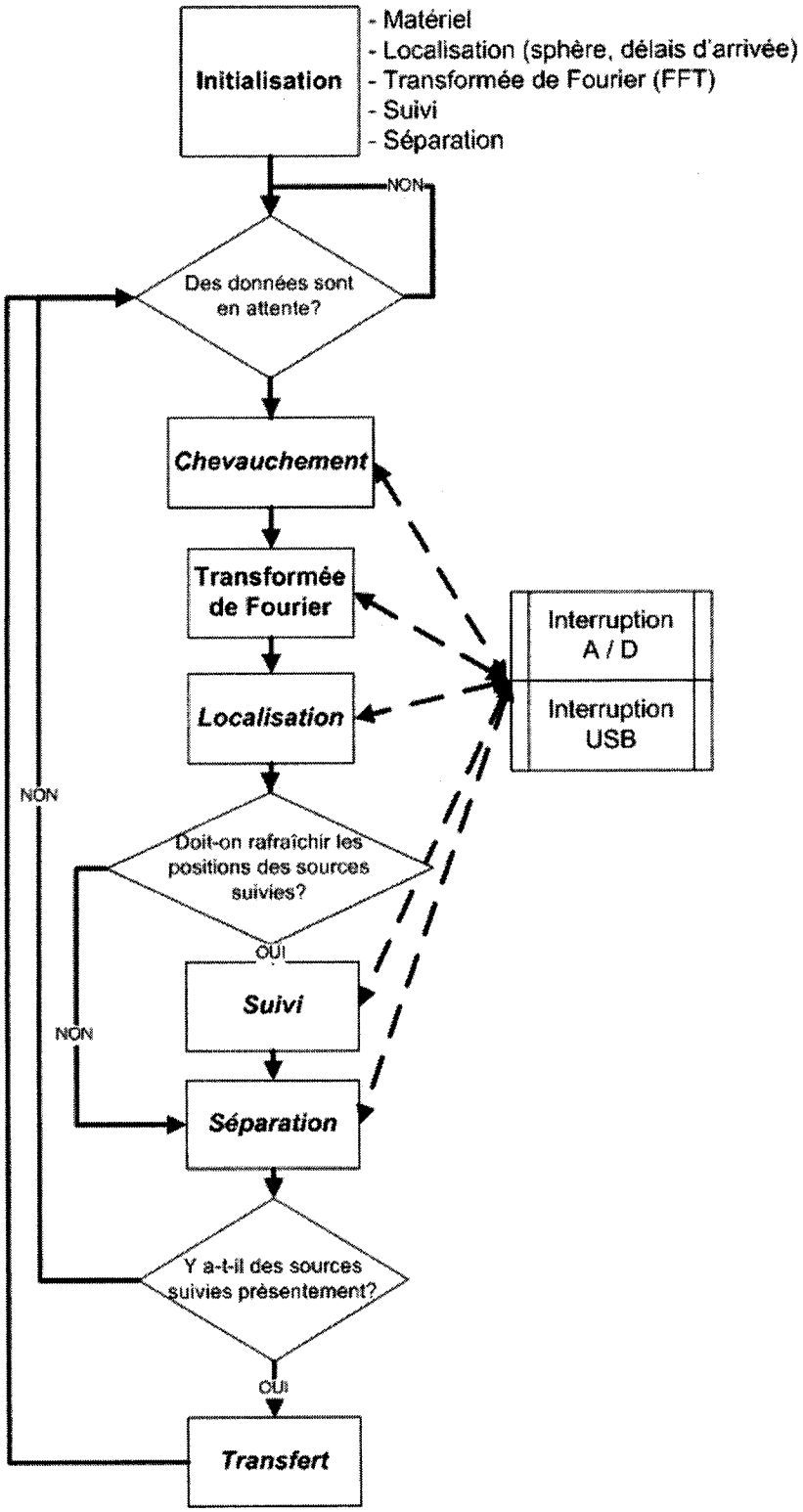

Figure 3.2 – Flux de données de l'architecture logicielle du système DSP

TABLEAU 3.2 – Détail de la section "variables et constantes" de la mémoire interne

Élément	**Taille(octets)**	**Justification**
Échantillons actuels des microphones	32 768	Ces données étant transférées par EDMA, elles doivent être dans la mémoire interne.
Tampon de calcul général	8192	Ce tampon est utilisé pour accélérer des calculs dans la localisation et le suivi. Il doit être en mémoire interne (et non sur la pile) puisque pour profiter des optimisations faites dans les librairies fournies par le DSP, il doit être aligné sur 4 octets.
Fenêtrage des données	4096	Ce tableau étant utilisé pour filtrer les données reçues et pour le sous-système de séparation, il est plus efficace de le mettre en mémoire locale.
Positions des microphones	96	Utilisés lors de l'initialisation, leur position en mémoire interne est principalement reliée à leur usage dans le sous-système de séparation.
Valeurs de métrique	42	Ces valeurs incluent le nombre d'itérations, le nombre de trames perdues, le nombre de trames en attente, le nombre de points dans la sphère de localisation , des pointeurs vers la position de lecture et d'écriture des tampons d'entrées et le nombre de sources suivies et localisées. Ce sont des données utilisées fréquemment et donc qui méritent d'être en mémoire interne.

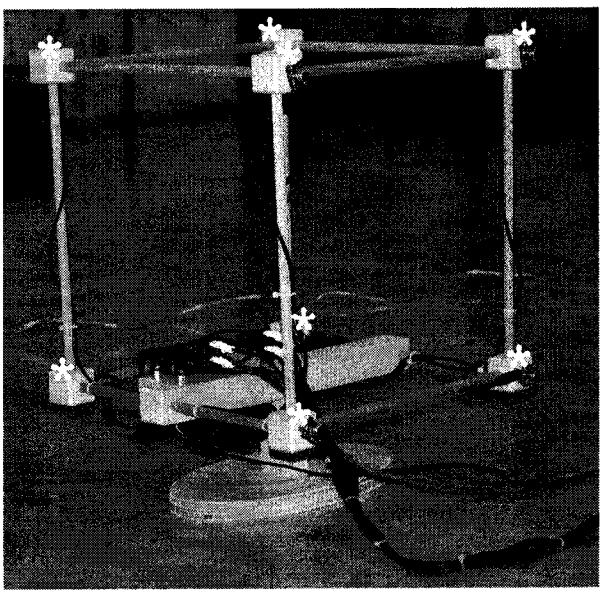

Figure 3.3 – Configuration physique des microphones utilisés pour les tests

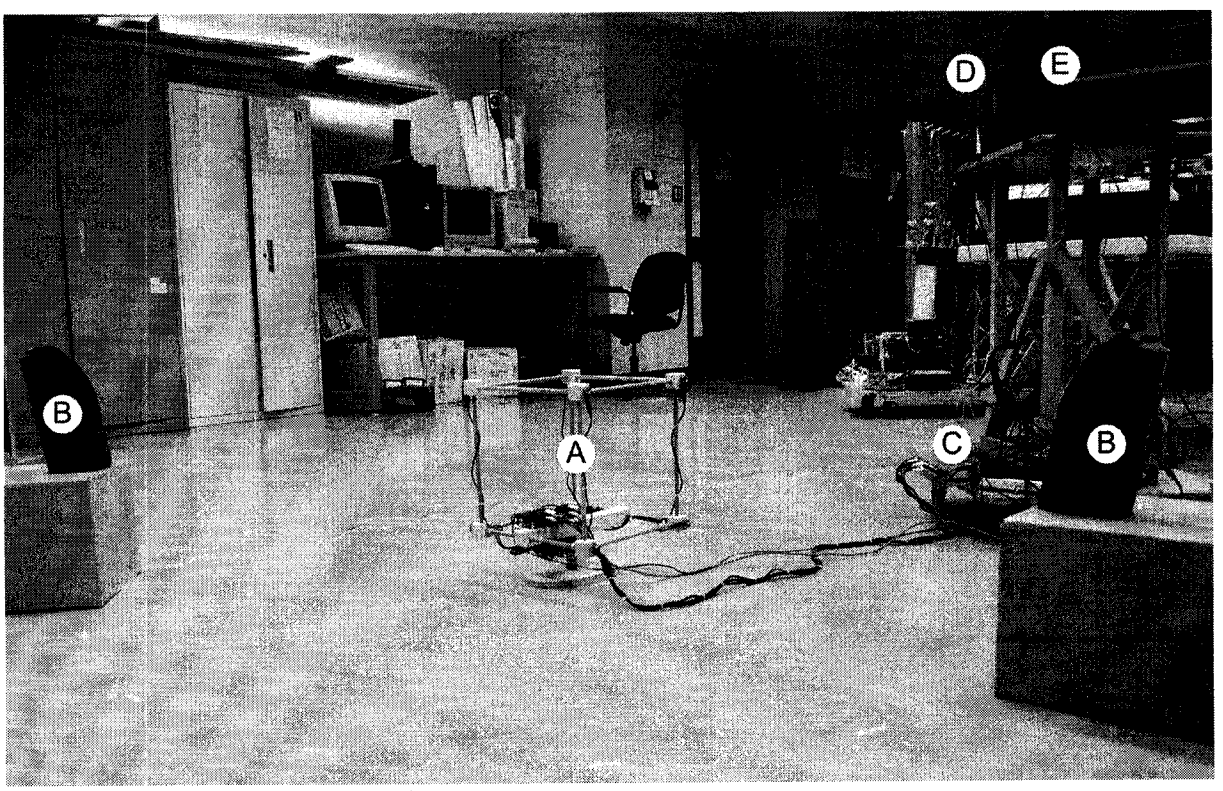

Figure 3.4 – Montage expérimental des tests

CHAPITRE 4

ARTICLE

Le corps de ce mémoire est constitué d'un article qui a été soumis dans le cadre de la conférence IROS (*International Conference on Intelligent Robots and Systems*) 2007. L'article présente les résultats obtenus dans l'implémentation du système sur DSP. L'article est écrit par Simon Brière (premier auteur), Jean-Marc Valin (l'auteur du système original) François Michaud et Dominic Létourneau.

4.1 Titre et résumé

Intégration d'un système auditif pour robot mobile

Résumé

AUDIBLE est le nom de notre système d'audition artificiel, capable de localiser plusieurs sources sonores simultanées, de les suivre et de les séparer en temps réel. Fonctionnant sur un ordinateur portable (Pentium M 1.6 GHz) et utilisant une carte audio *RME Hammerfal DSP Multiface* pour échantillonner huit microphones, le système est capable de localiser et de suivre jusqu'à quatre sources tout en séparant jusqu'à trois sources en temps réel, dans des environnements bruités. Pour utiliser ce système sur des robots et des systèmes dédiés qui n'ont pas autant de puissance de calcul de disponible, le système a été transféré sur un processeur de type DSP. Cet article décrit les compromis qui ont dus être réalisés pour concevoir l'implémentation embarquée d'AUDIBLE, de même que les performances observées. L'implémentation DSP (*Digital Signal Processor*) est entièrement fonctionnelle, avec quelques limitations par rapport au système original, soit des limitations sur la durée des sources sonores et la quantité de sources suivies simultanément.

4.2 Embedding of an Auditory System for a Mobile Robot

4.2.1 Abstract

AUDIBLE is how we named our artificial audition system, capable of sound source localization, tracking and separation in real time. Running on a laptop computer (1.6 GHz Pentium M) and using a RME Hammerfal DSP Multiface sound card to sample eight microphones, the system is able of localizing and tracking up to four sources, while separating up to three sources in real time in noisy environments. To use this system on robots and dedicated systems that do not have such processing power available, the system was ported to a DSP processor. This paper reports on the compromises made to design such embedded implementation of AUDIBLE, along with the observed performances. The DSP (Digital Signal Processor) implementation is fully functional and performs well with some limitations compared to the original system, i.e., limitations on sound source duration and on the number of sources that can be processed simultaneously.

4.2.2 Introduction

Localizing sound sources in our surroundings or understanding somebody talking while moving in a crowd are common in human interactions in real life. For a robot, however, such ability is not easily reproduced, having to deal with ambient noise and mixed sound sources. In the recent years, interest on artificial robotic audition has grown continuously, as it can be seen from the increasing number of robots exploiting such sense such as COG [6], SIG and SIG2 [26] and Spartacus [4, 23].

AUDIBLE is the name of the audition system used on Spartacus, developed to solve the problems of simultaneous sound sources localization, tracking and separation (SSLTS) [37, 38, 41]. The system works in real time using eight microphones, and is able to localize, track and separate simultaneous sound sources [36]. AUDIBLE was tested and successfully demonstrated in various environments, such as the AAAI 2005 [23] and 2006 Mobile Robot

competitions [24].

However, AUDIBLE is implemented on a regular laptop running Linux, and requires most of its processing power. With limited processing capabilities on a robot, AUDIBLE takes up resources that cannot be used for other robotic capabilities, such as vision. Adding a dedicated laptop for this requires space and energy, adds weight and increases cost, requirements that are not always easily met, especially for compact-size robots used for instance to study human-robot interaction with autistic children [25]. Therefore, a more compact, lighter, cheaper, low-power consumption solution would be beneficial. An embedded DSP (Digital Signal Processor) implementation of AUDIBLE would provide such advantages, as of making AUDIBLE platform-independent and facilitating integration on various robotic systems. Our long-term objective is to have a small, low-consumption DSP board with eight analog inputs capable of processing AUDIBLE's localization, tracking and separation algorithm.

However, because DSPs have limitations in terms of memory and processing power, our current work aims at identifying the compromises to be made in porting AUDIBLE on a DSP, as for demonstrating its capabilities compared to the original implementation. The paper briefly explains the original system, putting in perspective the design choices required to build a functional embedded version of AUDIBLE. It also presents the problems encountered in the process, the observed performance of our DSP implementation, and perspectives on how to improve this implementation.

4.2.3 Original AUDIBLE System

The AUDIBLE system, illustrated in Fig. 4.1, is composed of a sound source localization subsystem that detects, localizes and tracks sound sources in the environment, and a sound source separation subsystem that uses the localization information to separate each source. The sampling rate used in the original system is 48 kHz.

Sound Source Localization

The sound source localization component is described in [36, 38]. It consists of an initial localization step based on the steered response power algorithm and a tracking step that

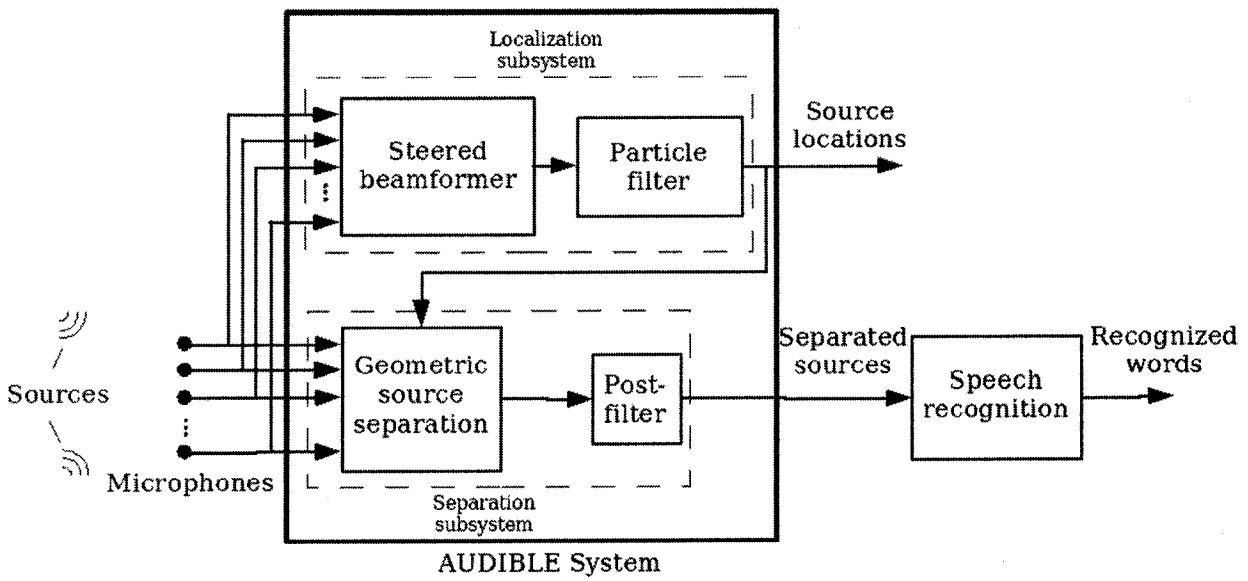

Figure 4.1 – Overview of AUDIBLE

is performed using particle filtering. For the steered response power algorithm, the source direction is initially searched on a 2562-point spherical grid. The direction can be searched efficiently using only $N(N-1)/2$ sums per grid point :

$$direction = \underset{d}{\operatorname{argmax}} \sum_{i=0}^{N-1} \sum_{j=0}^{i-1} R_{i,j} \left(\text{lookup}_{i,j} [d] \right) \tag{4.1}$$

where $\text{lookup}_{i,j}[d]$ is a lookup table that returns the time delay of arrival TDOA between microphones i and j for the searched direction d and $R_{i,j}$ is the relevance-weighted phase transform (RWPHAT) [37, 38]. The latter is computed as

$$R_{ij}(\tau) = \sum_{k=0}^{L-1} \frac{\zeta_i(k) X_i(k) \zeta_j(k) X_j(k)^*}{|X_i(k)|\,|X_j(k)|} e^{j2\pi k\tau/L} \tag{4.2}$$

where $\zeta_i(k)$ is the Wiener gain for frequency k that takes into account both the noise and reverberation. Once a sound source is found using (4.1), it is possible to find subsequent sources, by forcing

$$R_{i,j} \left(\text{lookup}_{i,j} [direction] \right) = 0, \forall i, \forall j \tag{4.3}$$

The search process is repeated to find a preset number of sources, which leads to false detections when fewer sources are present.

The search in (4.1) is based on the far-field assumption (large distance to the array) with a grid that provides a maximum error of 2.5° (best case), which corresponds to the radius covered by each of the 2562 regions around its center.

It is however possible to improve the resolution by performing a refined search, constrained to the neighborhood of the first result found. In this second search, we can include the distance. While this distance estimate is not reliable enough to be useful, it helps improve the direction accuracy. In addition to the refining stage, most floor reflections can be eliminated by having the search exploit the fact that a reflection always has the same azimuth as the direct path, but with a higher absolute elevation.

The direction information found by the steered beamformer generally contains a large number of false positives and false negatives. Moreover, (4.1) is memoryless and is thus unable to keep track of sources over time, especially when there are *gaps* in the localization data for a source. For this reason, we use a particle filtering stage. The choice of particle filtering is motivated by the fact that taking into account false positives and false negatives makes error statistics depart significantly from the Gaussian model. Each source being tracked is assigned a particle filter and the each observed direction in (4.1) is assigned to a tracked source using a probabilistic model [38]. By using the simple sample importance resampling (SIR) algorithm, it is possible to use 1000 particles per source while maintaining a reasonable complexity.

Sound Source Separation

The sound source separation subsystem [36, 41] is also composed of a linear sound source separation algorithm, followed by a non-linear post-filter. The initial linear source separation is achieved using a variant of the Geometric Source Separation (GSS) algorithm [28] that operates in real time and with reduced complexity [41].

The GSS algorithm alone cannot completely attenuate the noise and interference from other sources, so a multi-source post-filter is used to further improve the signals of interest. The

post-filter is based on the short-term spectral amplitude estimator approach originally proposed by Ephraim and Malah [11]. Unlike the classical algorithm, the noise estimate used is the sum of two terms : stationary background noise and interference from other sources. The interference term is computed by assuming a constant leakage from the other sources [41].

4.2.4 Embedded System (AUDIBLE-DSP)

Hardware

The first task in porting the original system consists of selecting the embedded platform. Standard control processors (like PICs from Microchip) do not have enough computational power to process AUDIBLE's algorithm, and computer processors require a lot of electrical power to work. On the other hand, FPGA (Field-Programmable Gate Array) can be used to implement parallel algorithms, but it is hard to estimate the number of gates required for our SSLTS system, and the cost quickly increases with a large number of gates. Therefore, the more promising option for this first embedded implementation is to use processors designed specifically for signal processing, i.e., DSP.

Because AUDIBLE's algorithm uses a lot of floating-point operations (for Fast Fourier Transforms, probabilities, etc.), we chose to use a floating-point DSP, more specifically the TMS320C6713 Texas Instruments DSP. Using a fixed-point DSP is also possible, but would require more time to adapt the code for the processor. The TMS32C6713 is a 225 MHz floating-point processor with 256 kBytes of internal RAM memory and L1 and L2 cache support. According to the specifications, the processor is rated at 1800 MIPS and 1350 MFLOPS, and its architecture is optimized for audio processing, providing a bus to quickly transfer data between memory and external interfaces.

To capture the signals coming from the microphones, synchronized eight-channel analog-to-digital converters (ADC) are required to provide aligned audio frames. A communication interface is also required to transfer the processed data to a host system, typically a different computer on a robot. The communication interface has to be independent of the operating system and as standard as possible, for easy interconnectivity of the DSP system with other

computing systems. With all these considerations in mind, we chose to use a Lyrtech[1] CPA-II board, as shown in Fig. 4.2. This board has 24 bits analog-to-digital converters supporting sampling frequencies from 32 kHz to 192 kHz – this is sufficient considering that the original AUDIBLE system runs at 48 kHz. The board also provides 64 Mbytes of external memory (SDRAM, running at 100 MHz) and 8 Mbytes of flash ROM memory. It has a USB2 interface that provides the communication channel needed to transfer the processed data on the host system. The size of the CPA-II board is not an issue at this point, since we can design a smaller board once the software development on the DSP is completed.

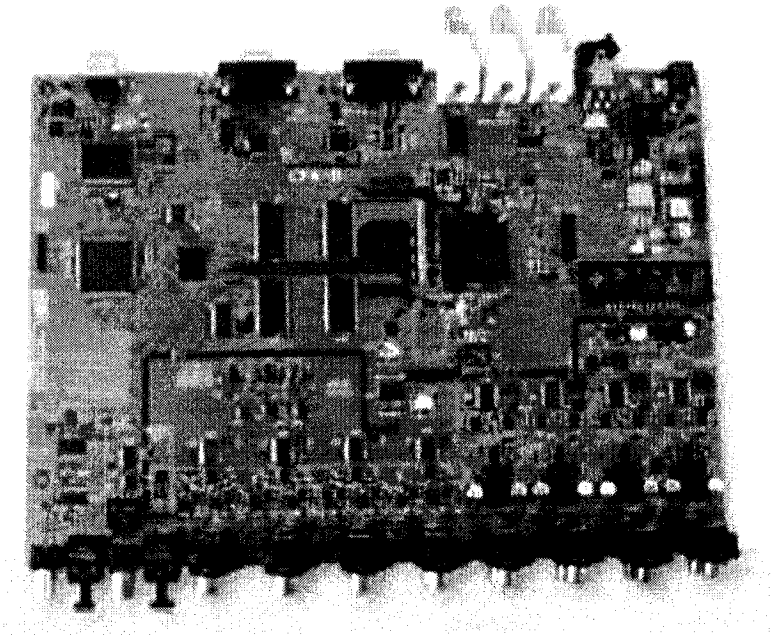

Figure 4.2 – Lyrtech CPA-II, the board used in our DSP development.

Porting AUDIBLE on a DSP

The first step toward porting the original AUDIBLE implementation to the DSP is to convert the original C++ code into C code, which is better supported by the DSP compiler. It is also necessary to remove dependencies to specialized libraries to carry out specific operations (e.g., FFTs), and find an equivalent way of implementing them on the DSP. Since the functions used in AUDIBLE are common in signal processing, this is done with a library that is included with the DSP.

[1]http ://www.lyrtech.com

The second step is to verify the accuracy of the code conversion. Because of the complexity of the original system and the change of some libraries, such verification is needed. We use pre-recorded microphone signals that are injected in the DSP using an emulator. At various stages of the algorithm, the data coming out is validated to ensure it is the same as the data processed by the original system.

The third step is to optimize the code for real-time processing. This is done by using optimized functions for the DSP and by modifying the loops to take advantage of the VLIW (Very Long Instruction Word) architecture that allowed to speed up parallel calculations. At this stage, it becomes apparent that memory management is a critical element on the DSP. Internal memory is fast but limited to 256 kbytes, and external memory is slow but very large (64 Mbytes). Since the algorithm uses a lot of tables (e.g., a 71736-bytes table is required to perform an accurate localization on a 2562 point grid around each microphone), it is impossible to fit all the code, the tables and the stack at the same time in the internal memory.

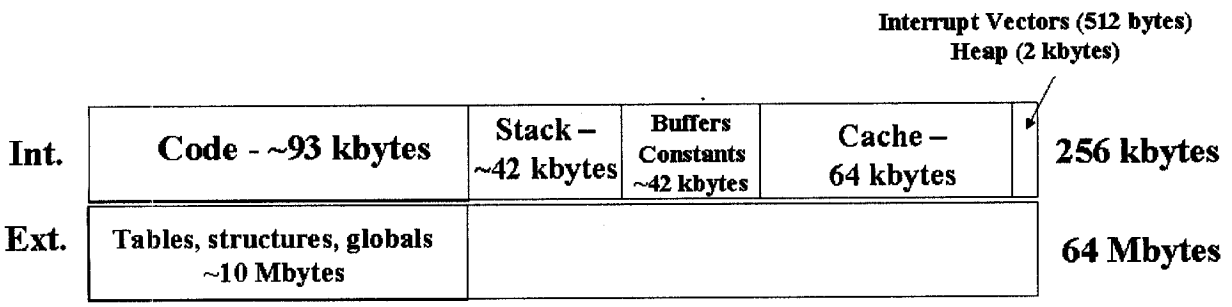

Figure 4.3 – Memory mapping of AUDIBLE-DSP.

The memory mapping used is shown in Fig. 4.3. L2 cache (64 kbytes) is enabled to accelerate repeated external memory access. The memory section containing the program instructions (code section in the figure, around 93 kbytes) could not be moved to the external memory, because doing so would considerably slow down the system, the processor having to make an external request at each instruction. Because of the structure of the system, a large stack (around 42 kbytes) is used to allocate local variables. A section of the internal memory (around 42 kbytes) is reserved for general temporary buffers to speed some sections of the code. A small section of the internal memory is reserved for the interrupt vectors (512 bytes)

and for the heap (2 kbytes). The external memory is mostly used as an audio buffer, large variables and for the large tables required by the algorithm, which currently uses around 10 Mbytes.

Because external memory is needed to store large tables that are accessed randomly – and thus cannot be properly cached – and because code optimization was done at the C level rather than at the assembly language level, the DSP implementation could not meet the same real-time performance of the original system, i.e., processing up to four sources at the same time at a sampling rate at 48 kHz. To allow the DSP implementation to process audio streams in real-time, the following modifications had to be made :

- *Sampling rate*

  In the original system, a sampling rate of 48 kHz was used. Using a 50% overlap for the separation subsystem and a frame size of 1024 samples, the processing is done in under 10.66 ms. In the DSP implementation, the sampling rate is lowered to 32 kHz, giving 16 ms for the maximum processing time between two 1024 samples frames with a 50% overlap.
- *Number of localized and separated sources*

  The number of localized and separated sources is brought down to two instead of the original value of four.
- *Directional refining*

  In the original system, a direction refining process is done when a source is found, as described in [38] and in Section 4.2.3. This requires extensive calculations, and has been removed from the DSP implementation.
- *Particle filters*

  The number of particles used in the particle filters is reduced empirically to 500 instead of the 1000 used in the original system.
- *Buffering*

  To be able to keep up with real-time constraints even when the processing time was over 16 ms, we now use a super-frame technique that mainly consist of buffering frames and to process them when there is time. In the current implementation, a buffer of 200 frames is used. If, however, there is currently no sources being tracked and separated and the

number of buffered frames gets over a threshold set to 25, the buffer is flushed. This is done in order to prevent a too large delay in the responsiveness of the system.

- *Position refreshing*
  In the original system, the positions of the sources were refreshed every 4 frames. This is a costly operation in terms of computational processing, and it is thus reduced to once every 5 frames in the DSP implementation.

The fourth step of the development process is to integrate the hardware part of the board, i.e., the interface to the ADC and the USB2 port, and the interruption management in the critical parts of the system. Interrupt management is needed since most of the optimized code used in the libraries and in tight loops is not interruptible.

4.2.5 Results

To correctly characterize the DSP implementation, we have to test each subsystem of AUDIBLE : localization, tracking and separation. We also have to collect information on the processing time of each subsystem in order to identify time-critical portion of the DSP implementation for future optimization. Another important aspect to evaluate is the reliability of the detection and localization of sound sources, and separation performances. All tests are done using the original system parameters, with no optimization of the implementation's parameters for the specific test cases.

The experimental setup is shown in Fig. 4.4. Since some of the tests involve recorded sounds, an amplifier and two speakers positioned around the microphone array are used as sound sources. The microphone array is mounted on a cube, and each microphone is attached to one corner of the cube. Each microphone is installed on a small PCB (Printed Circuit Board) with a configurable gain. This gain is adjusted so that each microphone has the same amplitude with a given reference signal. The signal from each microphone is connected both to the ADC of the DSP board and to a capture card installed on a laptop.

Tests are conducted in a typical lab environment with people working. There is noise coming from the ventilation system and background noise from chairs, computers and printer. Laptop

1 runs the original AUDIBLE system, while Laptop 2 serves as a client system for the DSP. Laptop 2 is connected to the DSP using USB2. Each laptop records the reported sources position over time and each separated stream. Comparison of the two systems is possible because both are connected to the same microphone array. Each system having its own capture board, there is a different level of noise added in the signals during the sampling process. It is however assumed that this noise is negligible compared to environmental noise, making the differences observed between the original AUDIBLE system and the DSP system attributable only to the setup that produced the results.

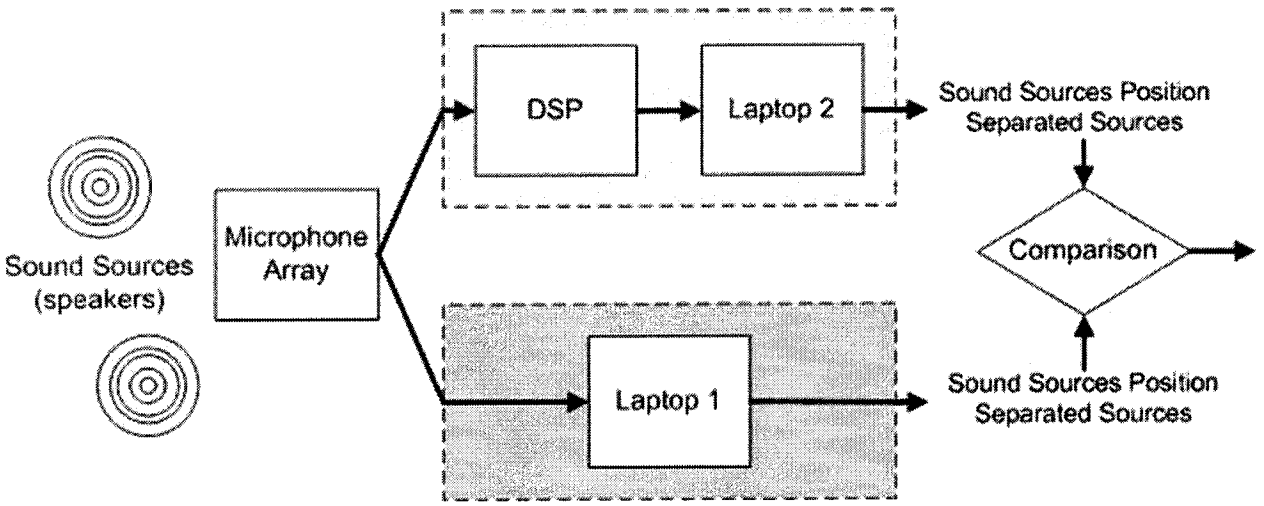

Figure 4.4 – Diagram of our experimental setup.

Processing Time

The first test on AUDIBLE-DSP is a time calculation for each of the subsystems in the algorithm. The timings are calculated using the internal DSP timer, averaged over a 5-second period. The results are shown in Table 4.1. "Source" refers to a source being separated, while "filter" refers to a source being tracked. The "Best Case" time refers to the moment when the localization positions are not being refreshed (4 out of 5 frames). The "Worst Case" time refers to the moment when the positions are being refreshed (1 out of 5 frames). Some states are not possible and are not displayed in the table. The Idle time (t_{idle}) is defined as the amount of time the system is not doing anything over a 80 ms period (5 frames). The objective is to have a $t_{idle} > 0$, since if the time is negative, the system has to buffer the frames for later processing, thus increasing the latency of the system. The maximum time

TABLEAU 4.1 – Processing time of AUDIBLE-DSP

Status	Best Case (ms)	Worst Case (ms)	t_{idle} (ms)	$t_{overflow}$ (s)
0 source, 0 filter	8.5	25.6	20.4	∞
0 source, 1 filter	8.5	29.1	16.9	∞
0 source, 2 filters	8.5	33.3	12.7	∞
1 source, 1 filter	12.2	32.8	**-1.6**	64
1 source, 2 filters	12.2	37.0	**-5.8**	17.7
2 sources, 2 filters	15.0	39.8	**-19.8**	5.2

before frame dropping ($t_{overflow}$) can be calculated using (4.4) :

$$t_{overflow} = \frac{Buf \cdot L \cdot t_{max}}{|t_{idle}| \cdot f_s} \tag{4.4}$$

where Buf is the buffer size (200 frames), L is the frame length (1024 samples) and f_s is the sampling rate (32 kHz).

According to these results, the DSP system is able to process 64 seconds of speech without frame dropping when there is one source, but is only able to process 5.2 seconds with 2 sources. There is an increasing delay in the response of the system as the buffer fills, but the system is still able to process in real time. The results indicate that the DSP implementation is unable to handle lengthy speech or sound sources without having to drop frames, as reflected by the negative values of t_{idle} shown in the table 4.1. Therefore, the quality of the separation system and the precision of the position of the sound sources are affected. Using a bigger frame buffer would delay the overflow, but would increase system latency.

Note that these times cannot be compared to the timing of the original system. Since that system runs on Linux (a non-real time operating system), it is difficult to evaluate precise execution time of specific functions because there is no guarantee that a specific function will not be interrupted by the system scheduler.

Detection

TABLEAU 4.2 – Detection reliability of AUDIBLE

Sound	Original system	DSP
Hand clap	100%	65%
Voice	100%	100%
Noise	100%	100%

Only one loudspeaker is used for this test. We consider sound source detection to be reliable if the system can detect every sound sources in its vicinity and if it is able to roughly localize it with a precision of 10° at distance of 1 meter. The loudspeaker is positioned on a 1-meter radius circle centered in the middle of the microphones array. The loudspeaker is placed at a height of 46 cm from the center of the microphone array, which is the origin of the positions reported by the localization system. The circle is divided into 16 equal sections of 22.5° each, starting at 0°. A pre-recorded audio stream consisting of 30 sounds is then played by the loudspeaker in each of the 16 positions on the circle. The audio stream is made of three types of sounds : hand clap, voice command (2 sec) and white noise burse (100 msec). Ten samples of these sounds occurring in sequence make the test stream.

Table 4.2 summarizes the results. The original system obtains a perfect score for the detection of each sound type. The DSP implementation also gets 100% detection for the voice and noise sounds, but does not perform as well with hand claps. This is caused by the position refreshing rate which is set at 1 every 5 frames for AUDIBLE-DSP. This justification has been verified by setting the refresh rate to the original value, bringing back a perfect score in detecting hand claps. Such performance can be considered acceptable since AUDIBLE would be mostly used to process speech and longer sound sources.

Localization

The objective here is to compare the accuracy of the two implementations. Using the same test setup of Section 4.2.5, two measures are taken : the accuracy of the azimuth angle of the detected sources, and the accuracy of the elevation. The root mean squared error is calculated by evaluating the difference between the angles returned by the DSP implementation and the original system.

TABLEAU 4.3 – Localization accuracy of AUDIBLE-DSP system (expressed by the root mean squared error with the original system)

Sound	Azimuth	Elevation
Hand clap	1.8°	2.1°
Voice	1.9°	2.6°
Noise	2.2°	2.9°
Overall	2.0°	2.6°

The results are shown in Table 4.3. Considering that the original system accuracy is around 1.1° (azimuth) and around 0.89° (elevation) [36] in a similar environment, the global error of the DSP implementation can be estimated as 3.1° (azimuth) and 3.5° (elevation). The difference between the accuracy of the two systems comes mainly from the removal of the direction refining phase from the DSP implementation. By doing so, processing time is reduced, but accuracy is also reduced. Nonetheless, the accuracy obtained is sufficient for most applications and is also similar to the accuracy of the human ear [31], which ranges between 2° and 4° in similar conditions.

Tracking

In this test, instead of using a static loudspeaker, two persons are asked to walk on a 2-meter radius circle around the microphones array, walking at normal speed and reading standard French text at normal pace. The tracking testing is done in two phases. In the first phase, the persons start at a precise position (90° for the first person and −90° for the second one), walk 90° to their right and then 180° to their left. This allows us to find the accuracy of the tracking in the case where the sound sources are not crossing. In the second phase, the persons start at a precise position (180° and 0°) and one walk 180° to the left and the other 180° to the right, crossing at 0°.

The resulting paths are shown in Fig. 4.5. Person 1 is in blue, Person 2 is in green. Naturally, the trajectories tracked by the original system are smoother because the localization refresh rate is greater. However, discrepancies are seen in the DSP trajectories. At the crossing point, the DSP implementation also seems to confuse sound sources for a short time. These are probably caused by the reduction of the number of particles in the filters and the removal of

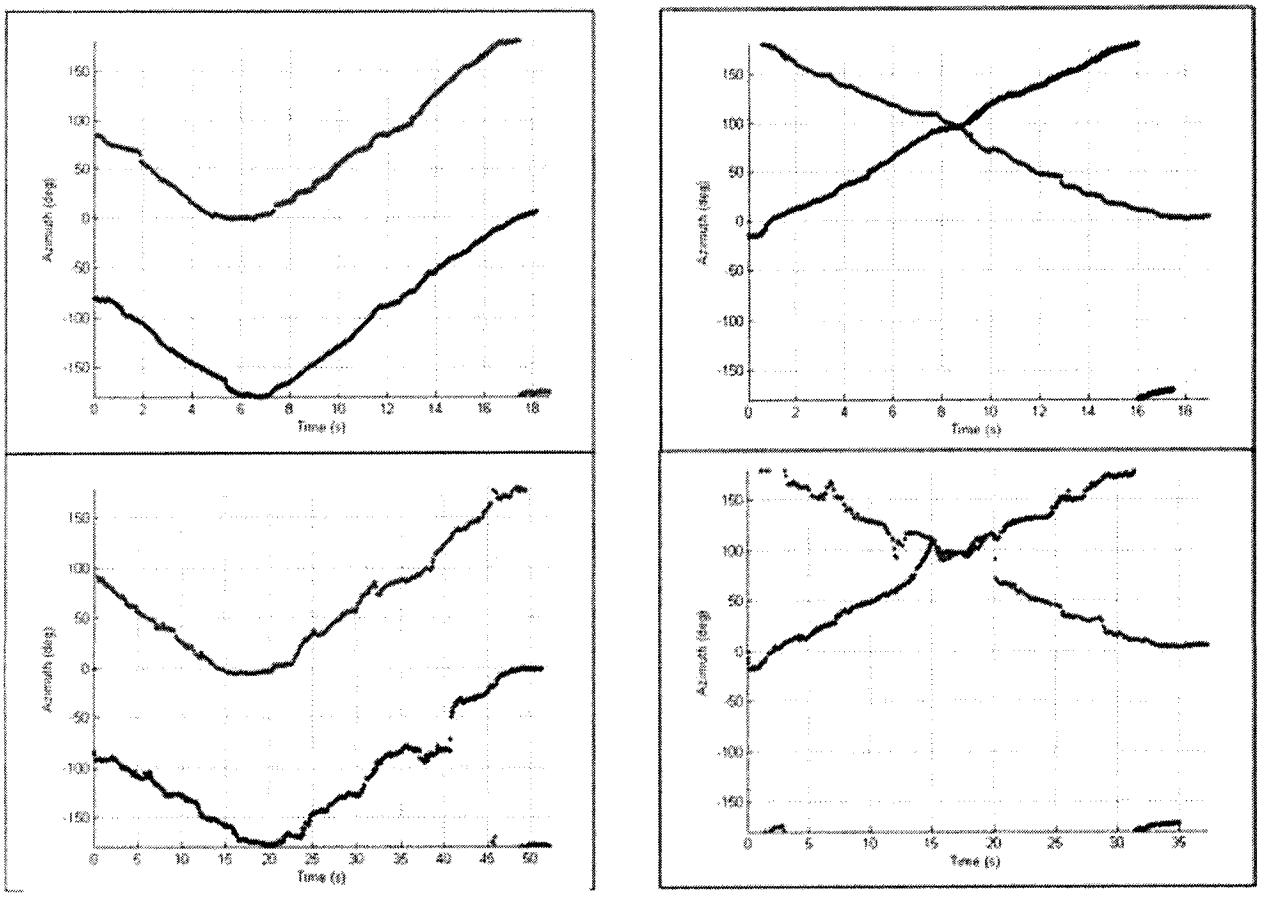

Figure 4.5 – Tracking results for the original system (top) and the DSP implementation (bottom) a) speakers not crossing; b) speakers crossing paths.

the direction refining phase in the localization subsystem. However, even if the paths from the DSP implementation are less precise, the tracking is efficient because both speakers can clearly be tracked.

Separation

To characterize the separation subsystem, two fixed loudspeakers were placed at the following locations : 0° and −90°, −90° and 135°, 0° and 135°. Three trials were done with a stereo recording made of 100 four-digit strings spoken by a mix of different speakers (half are women, half are men). The digits are taken from the AURORA database [29] and are mixed so that they are generated at the same time, one from the left speaker and one from the right speaker. The audio streams separated by AUDIBLE (original and DSP) are then sent to NUANCE[2], without any post-processing. Performance of the systems is based on the recognition rate by

[2]http ://www.nuance.com

TABLEAU 4.4 – Recognition accuracy of four-digits numbers separated by the systems

Result	Original System	DSP
Correct	56%	46%
Incorrect	3.3%	5.3%
Rejected	26.3%	36.7%
Not Captured	8.3%	6.3%
Incomplete	6.0%	5.7%

NUANCE of the spoken digits. The speech recognition accuracy of NUANCE on the clean (not corrupted) digits from the database is 100%.

Note that this test is not done to determine the performance of AUDIBLE in a speech recognition context, but for the sake of comparing the two implementations on the same basis. Streams coming out of the original AUDIBLE system has a recognition rate of around 90% for two speakers [36] when their localization is known. Original AUDIBLE system shows a lower rate in this test condition because the recordings are sampled at 8 kHz while AUDIBLE is optimized for sampling rate over 20 kHz.

Table 4.4 shows the average results on the four-digit strings over the following five categories : correctly recognized, incorrectly recognized (from one to three incorrect digits), rejected (no digit recognized), not captured (digits that were played, but not captured by the SSLTS system) or incompletely formed (there was less than four digits captured by the system). The objective is therefore to maximize correctly recognized strings while minimizing the other categories. Obviously, we can notice a lower performance in correctly recognized strings from the DSP implementation compared to the original one. The 10% difference seems mainly to be coming from a larger rejection rate on streams produced by the DSP. There is an audible difference between the processed streams of each system, but the streams are all human understandable. Further adjustments of the parameters in the DSP system could improve that rate.

4.2.6 Conclusions and Future Works

This paper reports on our successful, fully functional DSP implementation of AUDIBLE. Even though it is more limited compared to the original implementation, it could certainly be usable in a short- command speech recognizer system on a robot. It is important to realize that even with many approximations on the system, similar results are achieved with the DSP implementation, with faster code. The implementation is thus a good first step toward a faster and more compact system.

Further works on the embedded system will aim at improving the performances of the system. Surely, a floating-point DSP with a larger internal memory would be beneficial. But now that we have a first embedded implementation, it may be worth investigating the combination DSP/FPGA, or even only the use of a FPGA, for improving processing speed and capabilities. Another option is to transfer the system on a fixed-point DSP to take advantage of lower power consumption, lower cost, higher internal clock and larger internal memory. The main underlying objective of such improvements is to be able to easily benefit from the advantages of hearing on all kinds of robots and systems operating in the real world.

4.2.7 Acknowledgments

F. Michaud holds the Canada Research Chair (CRC) in Mobile Robotics and Autonomous Intelligent Systems. Support for this work is provided by the Natural Sciences and Engineering Research Council of Canada, the Fonds Québecois de la Recherche sur la Nature et les Technologies, the Canada Research Chair program and the Canadian Foundation for Innovation.

4.3 En résumé

L'article présenté dans ce chapitre présente les résultats et les compromis qui ont dû être effectués dans le transfert du système original (ordinateur) vers un système embarqué (DSP). Ces compromis proviennent principalement de délais occasionnés par des accès mémoires et

par le fait que les optimisations logicielles se situent à un langage de programmation de plus haut niveau (C, comparativement à un langage assembleur par exemple). Les résultats obtenus montrent que le système fonctionne bien en temps réel, malgré ces limitations et les compromis qui en découlent.

CHAPITRE 5

COMPLÉMENT SUR LES TESTS DU SYSTÈME

Les tests présentés dans l'article pour les délais, la localisation et le suivi de sources s'avèrent suffisants pour caractériser le système. Cependant, les résultats présentés pour la séparation ne représentent pas bien les performances réelles du système, et des tests supplémentaires ont été réalisés pour pallier à cette situation. Ce chapitre présente les résultats des tests supplémentaires qui ont été effectués sur le sous-système de séparation.

Premièrement, les paramètres du système de reconnaissance vocale (NUANCE) ont été ajustés en fonction des paramètres présentés dans le tableau 5.1. Ces paramètres sont tirés du travail de Brosseau [7] et sont adaptés au système de localisation, suivi et séparation de sources. Pour fin de comparaison, tous les tests qui sont fait dans cette section utilisent ces nouveaux paramètres.

Deuxièmement, dans les tests effectués dans l'article, une reconnaissance par phrase était effectuée. Une phrase était définie comme étant une série de quatre chiffres. Cette approche représente bien ce qui est utilisé pour reconnaître une commande vocale. Cependant, la lacune principale de celle-ci est de ne pas reconnaître correctement une phrase si un des chiffres la composant n'est pas reconnaissable. Ceci a bien sûr comme effet de réduire grandement le taux de reconnaissance. Les tests effectués par Valin et Brosseau (présentés dans [36] et [7]) ont été effectués sur des chiffres et non sur des phrases. Afin de comparer les performances du système sur le DSP avec des résultats déjà connus, il faut donc prendre les mêmes paramètres.

Troisièmement, les échantillons sonores obtenus à partir d'AURORA sont des échantillons avec une fréquence d'échantillonnage à 8 kHz, soit la fréquence utilisée par la plupart des systèmes de reconnaissance vocale. Or, le système de localisation, suivi et séparation présenté dans ce mémoire est conçu pour fonctionner avec une bande audio plus large, soit d'environ

TABLEAU 5.1 – Paramètres du système de reconnaissance vocale NUANCE

Paramètre	**Valeur**	**Justification**
rec.ConfidenceRejectionThreshold	0	En fixant cette valeur à 0, on accepte tous les résultats fournis par NUANCE, même si le degré de confiance du système est faible. Le sous-système de localisation effectue déjà un filtrage du bruit et donc ce qui est envoyé à NUANCE est déjà reconnaissable. Défaut : 45
ep.VoiceThreshold	50	Cette valeur permet de considérer chacun des échantillons sonores comme étant une phrase entière. Défaut : 5
ep.ThresholdSnr	5	L'engin de NUANCE réagit plus rapidement. Ce n'est pas un paramètre critique pour ces tests, mais intéressant à régler pour accélérer les tests. Défaut : 14

20 kHz. En utilisant des échantillons à 8 kHz, le système a plus de difficultés à localiser correctement les sources. Comme la séparation est étroitement liée à la précision de la localisation, ce plus faible taux d'échantillonnage a un impact sur les taux de reconnaissance. Pour qualifier les performances du système, il faut donc utiliser des sources sonores qui représentent bien le contexte d'utilisation "réel" du système.

Trois tests ont donc été réalisés pour prendre en considération l'impact des éléments énoncés ci-dessus. Tout d'abord, les données enregistrées à partir des échantillons provenant de la base de données d'AURORA ont été utilisés avec le système de reconnaissance vocale. Une reconnaissance par phrase et par chiffre a été réalisée avec ces échantillons. Les échantillons originaux (i.e., qui sont présentés à l'entrée du système) obtiennent un taux de reconnaissance de 100% lorsqu'une reconnaissance par phrase est effectuée et un taux de reconnaissance de 100% lorsqu'une reconnaissance par chiffre est réalisée. Ensuite, les mêmes données ont été passées dans le système de reconnaissance, mais en utilisant cette fois-ci une reconnaissance par chiffre et non par phrase. Au total, 1200 chiffres ont été donnés en entrée aux deux systèmes. Finalement, des nouveaux échantillons ont été enregistrés pour être à leur tour

soumis aux deux systèmes dans les mêmes conditions. Les voix de sept femmes et de neuf hommes ont été enregistrés à une fréquence d'échantillonnage de 48 kHz. Ces personnes avaient à prononcer des chiffres assemblés par groupes de quatre afin de constituer une base de données similaire à celle utilisée pour la première série de tests. Tout comme les tests présentés dans l'article à la section 3.5, 50 phrases de quatre chiffres ont été assemblées dans un enregistrement. Les échantillons sonores utilisés dans cet enregistrement ont un taux de reconnaissance de 100% par NUANCE avant d'être présentés aux systèmes. Ils ont été reconnus par NUANCE configuré pour reconnaître les chiffres individuellement. Le système a donc reçu 50 phrases (25 échantillons d'hommes, 25 échantillons de femmes) de quatre chiffres sur deux haut-parleurs à trois positions différentes, pour un total de 1200 chiffres, et ce respectivement pour les tests de séparation effectués, avec les échantillons d'AURORA et avec les enregistrements à 48 kHz.

TABLEAU 5.2 – Résultats des tests supplémentaires du sous-système de séparation

Tests	**Taux de reconnaissance**					
	Système original			*DSP*		
	M	F	**Moyenne**	M	F	**Moyenne**
Phrases complètes (AURORA, paramètres originaux)	63%	49%	**56%**	53%	40%	**46%**
Phrases complètes (AURORA, paramètres ajustés)	73%	69%	**71%**	69%	63%	**66%**
Chiffres (AURORA, paramètres ajustés)	84%	80%	**82%**	83%	80%	**82%**
Chiffres (échantillons à 48 kHz)	95%	91%	**93%**	91%	88%	**90%**

On constate, dans les résultats de tests présentés dans le tableau 5.2, que tous ces changements ont eu pour effet d'améliorer les performances du système. Les pourcentages qui sont présentés dans le tableau représentent le taux de reconnaissance pour chacun des tests effectués, divisés par le type de phrase : homme (M), femme (F) et la moyenne des deux types. Il est à noter que la position des hauts-parleurs n'influence pas de façon significative (entre 1% et 5%) le taux de reconnaissance, et les données présentées correspondent donc à la moyenne des résultats de toutes les positions testées. Il est intéressant de constater qu'une

fois les paramètres de NUANCE ajustés, il n'y a pratiquement pas de différences entre les taux de reconnaissance des hommes et des femmes.

Le simple fait d'ajuster les paramètres de NUANCE a amélioré les performances des deux systèmes de près de 20%. Cette augmentation s'explique principalement par le paramètre qui fixe le taux de rejet à 0. Ainsi, les phrases reconnues ne sont plus rejetées par le système même si le taux de certitude de la reconnaissance est très faible.

De plus, une amélioration importante est apportée en réglant le système de reconnaissance pour qu'il reconnaisse les chiffres individuellement. On constate alors que le système sur le DSP se retrouve avec un taux de reconnaissance comparable au système original. Ceci peut s'expliquer par le fait qu'il y avait, lors des tests initiaux, un plus haut taux de phrases non-reconnues par le DSP. Or, pour qu'une phrase ne soit pas reconnue, il suffit parfois d'un seul chiffre erroné. Avec une reconnaissance par chiffres, il n'y pas de rejection lorsqu'un seul chiffre est erroné - les trois autres peuvent être bons et compter comme une bonne reconnaissance.

Finalement, les résultats du dernier tests prouvent bien que le système fonctionne mieux avec des échantillons sonores ayant une plus large bande. Les taux de reconnaissance obtenus avec les échantillons à 48 kHz se rapprochent des paramètres utilisés dans la réalité. On constate donc une amélioration de près de 10% entre le test similaire effectué avec les échantillons à 8 kHz. Donc, malgré les compromis qui ont dû être réalisés pour parvenir avec une mise en œuvre en temps réel fonctionnel sur le DSP, les performances démontrées par le système embarqué sont très proches du système original.

CHAPITRE 6

CONCLUSION

Le présent ouvrage présente la mise en œuvre d'un système auditif pour un robot mobile sur un processeur de type DSP. Un tel système existe déjà sur un ordinateur portable, mais des choix de conception ont été effectués pour parvenir à réaliser le système embarqué sur DSP.

Les principales étapes qui réalisées sont les suivantes : la sélection du processeur embarqué, le transfert du code existant vers le nouveau système, l'optimisation du système et la caractérisation de celui-ci avec des tests de performance.

Le système implémenté sur le DSP est complètement fonctionnel et traite les données en temps réel, mais possède certaines limitations par rapport au système original. Ainsi, le système embarqué sur DSP fonctionne à 32 kHz (au lieu de 48 kHz), permet de suivre jusqu'à deux sources à la fois (au lieu de quatre), permet de séparer deux sources (au lieu de trois) et les positions rapportées sont mises à jour moins fréquemment (soit une fois à toutes les 80 ms au lieu d'une fois toutes les 43 ms).

Les tests ont confirmé que le système s'avérait moins précis au niveau de la localisation que le système orignal. Le système embarqué est moins précis de 2.0° pour les angles d'azimut et de 2.6° pour les angles d'élévation par rapport au système original. Malgré tout, la précision de la localisation est acceptable et demeure tout de même meilleure que celle de l'oreille humaine (qui varie entre 2° et 4°). De plus, le suivi de sources est fonctionnel malgré les compromis apportées au système. Le système embarqué sur DSP est capable d'effectuer le suivi de sources qui ne se croisent pas et qui se croisent adéquatement. Du côté du sous-système de séparation, les résultats sont tout à fait comparables entre les deux systèmes. Le taux de reconnaissance se situe à 90% sur le DSP comparativement à 93% sur le système original. Au niveau des délais de traitement, certains sous-systèmes requièrent un temps de traitement élevé par rapport au temps de traitement maximal de chaque trame (16 ms à 32 kHz) et nécessitent donc l'usage de tampons. Ces tampons permettent de suivre le flot de

données, mais introduisent également une latence qui varie selon les conditions du système. Le système est capable de suivre et séparer des courts échantillons (variant entre 64 secondes et 5.2 secondes selon les conditions), ce qui n'empêche pas le système embarqué de servir à une reconnaissance de commandes vocales, qui sont généralement assez courtes.

C'est sans doute au niveau de la diminution des délais de traitement que des améliorations peuvent être apportées. Des optimisations ont été effectuées, mais les optimisations ont porté sur le code en C et non sur du code assembleur (où un autre niveau d'optimisation serait possible) et n'ont pas porté sur l'organisation de la structure interne de l'algorithme pour que celui-ci s'exécute plus rapidement. Par exemple, il serait sans doute possible de réécrire certaines parties de l'algorithme en modifiant l'imbrication des boucles pour que les délais d'exécution soient diminués.

Les prochaines étapes de développement du projet porteront sur l'amélioration des performances du système. Il est évident qu'un processeur DSP ayant des capacités de mémoire interne plus grandes serait souhaitable. Le transfert un DSP de type point fixe, qui offre typiquement une meilleure consommation électrique, un plus grande mémoire interne et une vitesse plus grande pourrait aussi être envisagée. Un système embarqué faisant usage d'un DSP et d'un FPGA pourrait être également une solution pour pallier aux délais rencontrés. Toutes ces améliorations potentielles pourraient permettre de faire bénéficier des avantages d'utiliser un système d'audition dans des systèmes de robot mobile, mais aussi dans d'autres systèmes comme la vidéoconférence.

Références

[1] P. Aarabi, Q.H. Wang, et M. Yeganegi. Integrated displacement tracking and sound localization. Dans *Proceedings IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004)*, volume 5, pages 937–940, 2004.

[2] M. S. Arulampalam, S. Maskell, N. Gordon, et T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. Dans *IEEE Transactions on Signal Processing*, volume 50(2), pages 174–188, 2002.

[3] C. Breazeal et L. Aryananda. Recognizing affective intent in robot directed speech. Dans *Autonomous Robots*, volume 12, pages 83–104, 2002.

[4] S. Brière, D. Létourneau, M. Fréchette, J.-M. Valin, et F.Michaud. Embedded and integration audition for a mobile robot. Dans *Proceedings AAAI Fall Symposium Workshop Aurally Informed Performance : Integrating Machine Listening and Auditory Presentation in Robotic Systems*, volume FS-06-01, pages 6–10, 2006.

[5] D. Brock et E. Martinson. Using the concept of auditory perspective taking to improve robotic speech presentations for individual human listeners. Dans *Proceedings AAAI Fall Symposium Workshop Aurally Informed Performance : Integrating Machine Listening and Auditory Presentation in Robotic Systems*, volume FS-06-01, pages 11–15, 2006.

[6] R. Brooks, C. Breazeal, M. Marjanovie, B. Scassellati, et M. Williamson. The Cog project : Building a humanoid robot. *Computation for Metaphors, Analogy, and Agents*, C. Nehaniv, Ed. Spriver-Verlag, :52–87, 1999.

[7] Y. Brosseau. Intégration de composants décisionnels sur Spartacus, un robot participant au AAAI Challenge 2005. Mémoire de maîtrise, Université de Sherbrooke, Département de génie électrique et informatique, mars 2007.

[8] C. Choi, D. Kong, J. Kim, et S. Bang. Speech enhancement and recognition using circular microphone array for service robots. Dans *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3516–3521, 2003.

[9] S. Chowdhury, M. Ahmadi, G.A. Jullien, et W.C. Miller. A MEMS implementation of an acoustical sensor array. Dans *2001 IEEE International Symposium on Circuits and Systems (ISCAS2001)*, volume 2, pages 273–276, 2001.

[10] I. Cohen et B. Berdugo. Speech enhancement for non-stationary noise environments. *Signal Processing*, vol. 81, no. 2 :2403–2418, 2001.

[11] Y. Ephraim et D. Malah. Speech enhancement using minimum mean-square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech and Signal Processing (ASSP)*, 32(6) :1109–1121, 1984.

[12] J. Fritsch, M. Kleinehagenbrock, S. Lang, G. A. Fink, et G. Sagerer. Audiovisual person tracking with a mobile robot. Dans *Proceedings International Conference on Intelligent Autonomous Systems*, pages 898–906, March 2004.

[13] D. Gatica-Perez, G. Lathoud, I. McCowan, J-M. Odobez, et D. Moore. Audio-visual speaker tracking with importance particle filters. Dans *Proceedings International Conference on Image Processing*, volume 3, pages 25–28, 2003.

[14] D. Giuliani, M. Omologo, et P. Svaizer. Talker localization and speech recognition using a microphone array and a cross-power spectrum phase analysis. Dans *Proceedings International Conference on Spoken Language Processing*, pages 1243–1246, 1994.

[15] A. Haasch, S. Hohenner, S. Hüwel, M. Kleinehagenbrock, S. Lang, I. Toptsis, G. A. Fink, J. Fritsch, B. Wrede, et G. Sagerer. BIRON - The Bielefeld Robot Companion. Dans *Proceedings International Workshop on Advances in Service Robotics*, pages 27–32, May 2004.

[16] J. D. Han, S. W. Zeng, K. Y. Tham, M. Badgero, et J. Weng. Dav : A humanoid robot platform for autonomous mental development. Dans *Proceedings 2nd International Conference on Development and Learning*, pages 73–81, MIT, Cambridge, MA, 2002. IEEE Computer Society Press.

[17] R. Irie. Robust sound localization : An application of an auditory perception system for a humanoid robot. Mémoire de maîtrise, MIT Department of Electrical Engineering and Computer Science, 1995.

[18] W. Kellermann. A self-steering digital microphone array. Dans *Proceedings International Workshop on Acoustic Echo Control*, pages 3581–3584, 1991.

[19] D. Létourneau, F. Michaud, J.-M. Valin, et C. Proulx. Textual message read by a mobile robot. Dans *In Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2724–2729, 2003.

[20] Q. Lin, E. Jan, et J. Flanagan. Microphone arrays and speaker identification. Dans *IEEE Transactions on Speech and Audio Processing*, volume 2, pages 622–629, 1994.

[21] A. Mahajan et M. Walworth. 3-D position sensing using the difference in the time-of-flights from a wave source to various receivers. Dans *IEEE Transactions on Robotics and Automation*, pages 91–94, 2001.

[22] Y. Matsusaka, T. Tojo, S. Kubota, K. Furukawa, D. Tamiya, K. Hayata, Y. Nakano, et T. Kobayashi. Multi-person conversation via multi-modal interface - A robot who communicate with multi-user. Dans *Proceedings Eurospeech 99*, volume 4, pages 1723–1726, 1999.

[23] F. Michaud, C. Côté, D. Letourneau, Y. Brosseau, J.-M. Valin, E. Beaudry, C. Raievsky, A. Ponchon, P. Moisan, P. Lepage, Y. Morin, F. Gagnon, P. Giguere, M.-A. Roux, S. Caron, P. Frenette, et F. Kabanza. Spartacus attending the 2005 AAAI conference. *Autonomous Robots, Special Issue on the AAAI Mobile Robot Competitions and Exhibition*, 22(4) :369–384, 2007. Autonomous Robots (Springer).

[24] F. Michaud, D. Létourneau, M. Frechette, E. Beaudry, et F. Kabanza. Spartacus, scientific robot reporter. Dans *Proceedings AAAI Mobile Robot Workshop*, 2006.

[25] F. Michaud, T. Salter, A. Duquette, et J.-F. Laplante. Perspectives on mobile robots used as tools for pediatric rehabilitation. *Assistive Technologies, Special Issue on Intelligent Systems in Pediatric Rehabilitation*, 19 :14–29, 2007.

[26] M. Murase, S. Yamamoto, J.-M. Valin, K. Nakadai, K. Yamada, Komatani K., T. Ogata, et H. G Okuno. Multiple moving speaker tracking by microphone array on mobile robot. Dans *Proceedings European Conference on Speech Communication and Technology (Interspeech)*, pages 249–252, 2005.

[27] M. Omologo et P. Svaizer. Acoustic event localization using a crosspower-spectrum

phase based technique. Dans *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages II–273–II–276, Adelaide, Austrailia, 1994.

[28] L. Parra et C. Alvino. Geometric source separation : merging convolutive source separation with geometric beamforming. *IEEE Transactions on Speech and Audio Processing*, 10(6) :352–362, 2002.

[29] D. Pearce. Developing the ETSI AURORA advanced distributed speech recognition frontend & what next. Dans *Proceedings IEEE Automatic Speech Recognition and Understanding Workshop*, pages 131–134, 2001.

[30] D. Rabinkin, R. Renomeron, A. Dahl, J. French, J. Flanagan, et M. Bianchi. A DSP implementation of source location using microphone arrays. Dans *Proceedings of The International Society for Optical Engineering (SPIE)*, pages 88–99, 1996.

[31] B. Rakerd et W. M. Hartmann. Localization of noise in a reverberant environment. Dans *Proceedings International Congress on Acoustics*, pages 414–422, 2004.

[32] S. Rennie, P. Aarabi, T. Kristjansson, B. Frey, et K. Achan. Robust variational speech separation using fewer microphones than speakers. Dans *Proceedings IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 741–744, Hong Kong, April 2003.

[33] H. Silverman. A digital processing system for source location and sound capture by large microphone arrays. Dans *Proceedings of IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume I, pages 251–254, 1997.

[34] A. Stoytchev et R. C. Arkin. Incorporating motivation in a hybrid robot architecture. Dans *Journal of Advanced Computational Intelligence and Intelligent Informatics*, volume 8 No.3, pages 100–105, 2004.

[35] I. Toshima, S. Aoki, et T. Hirahara. An acoustical tele-presence robot : Telehead II. Dans *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2105–2110, September-October 2004.

[36] J.-M. Valin. *Auditory System For a Mobile Robot.* Thèse de doctorat, Département de génie électrique et informatique de l'Université de Sherbrooke, août 2005.

[37] J.-M. Valin, F. Michaud, et J. Rouat. Robust 3D localization and tracking of sound sources using beamforming and particle filtering. Dans *Proceedings IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 221–224, 2006.

[38] J.-M. Valin, F. Michaud, et J. Rouat. Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering. *Robotics and Autonomous Systems*, 55(3) :216–228, 2007.

[39] J.-M. Valin, F. Michaud, J. Rouat, et D. Letourneau. Robust sound source localization using a microphone array on a mobile robot. Dans *Proceedings International Conference on Intelligent Robots and Systems*, volume 2, pages 1228–1233, 2003.

[40] J.-M. Valin, J. Rouat, et F. Michaud. Enhanced robot audition based on microphone array source separation with post-filter. Dans *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3(28), pages 2123–2128, 2004.

[41] J.-M. Valin, J. Rouat, et F. Michaud. Microphone array post-filter for separation of simultaneous non-stationary sources. Dans *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 222–224, 2004.

[42] D.B. Ward, E.A. Lehmann, et R.C. Wiliamson. Particle filtering algorithms for tracking an acoustic source in a reverberant environment. Dans *Proceedings IEEE International Conference en Acoustics, Speech and Signal Processing*, volume 11, no. 6, 2003.

[43] S. Yamamoto, J.-M. Valin, K. Nakadai, J. Rouat, F. Michaud, T. Ogata, et H. G. Okuno. Enhanced robot speech recognition based on microphone array source separation and missing feature theory. Dans *Proceedings IEEE International Conference on Robotics and Automation*, pages 1477–1482, 2005.

[44] Y. Zhang et J. Weng. Grounded auditory development by a developmental robot. Dans *Proceedings INNS/IEEE International Joint Conference of Neural Networks*, pages 1059–1064, 2001.