SETIT 2005
3rd International Conference: **S**ciences of **E**lectronic, **T**echnologies of **I**nformation and **T**elecommunications
March 27-31, 2005 – TUNISIA

Utilisation des Réseaux de Petri Architecturaux pour la modélisation des algorithmes de commande d'une plateforme technologique d'aide aux handicapés

A. Abellard et M. Ben Khelifa

Laboratoire STIC-AI, Université du Sud-Toulon-Var, BP 20132, 83957 La Garde Cedex, France
alexandre.abellard@univ-tln.fr
khelifa@univ-tln.fr

Résumé: Pour évaluer les capacités fonctionnelles d'une personne handicapée et les compensations nécessaires à la commande d'un fauteuil roulant électrique, une plateforme technologique a été réalisée sur la base des conseils d'un ergothérapeute. Elle a été réalisée avec du matériel standard, comme il est souhaitable de le faire chaque fois que c'est possible pour réduire les coûts financiers. Différentes commandes ont été implantées et sont personnalisables en fonction des spécificités de chaque personne. Elles sont faciles à installer car elles sont modulaires et interchangeables. L'électronique embarquée est intégrée dans des circuits programmables de type FPGA. La programmation est faite en langage VHDL et est obtenue directement à partir de la modélisation des algorithmes par Réseaux de Petri Architecturaux pour obtenir une adéquation algorithme architecture optimisée.
Mots clés: Adéquation Algorithme Architecture, Fauteuil Roulant, Handicap, Réseaux de Petri.

1 Introduction

L'amélioration constante des performances des composants programmables permet de réaliser des circuits de commande très performants. Toutefois, pour obtenir une implantation optimisée des algorithmes, il est important de pouvoir disposer d'un modèle de représentation des opérations à exécuter. Ainsi la modélisation du parallélisme intrinsèque d'un algorithme de traitement du signal ou d'image pourra permettre de rechercher le meilleur compromis entre le coût en matériel nécessaire (et donc le coût financier) et le temps d'exécution (et donc la rapidité). En fonction du cahier des charges, une attention particulière devra être accordée à ce paramètre. C'est le cas par exemple de l'application développée au laboratoire et qui sera décrite en fin d'article à propos d'une commande temps – réel d'un fauteuil roulant électrique par le traitement d'images de la tête de l'utilisateur handicapé.

Le problème de l'Adéquation Algorithme Architecture a été très étudié par la communauté scientifique en utilisant différents modèles malheureusement souvent liés à un type de matériel. Pour s'affranchir de cette contrainte et pour mettre au point un outil standard totalement indépendant des types de composants, nous avons défini un modèle reposant sur une extension des Réseaux de Petri (Diaz, 2001) que nous avons appelé Réseaux de Petri Architecturaux (RdPA).

Leur intérêt majeur réside dans l'unification des flots de données et des flots de contrôle. Ils permettent de réaliser toutes les étapes d'analyse, de simulation et de validation d'un algorihme ainsi que l'obtention automatique d'un programme VHDL pouvant être utilisé avec des composants programmables comme par exemple les circuits FPGA.

Les RdPA sont consitués de deux parties : les Réseaux de Petri à Flux de Données (RdPFD) et les Réseaux de Petri à Flux de Contrôle (RdPFC). Les RdPFD utilisent le concept de frontière de factorisation pour optimiser la répétition de sous-réseaux identiques en traitant les données par énumération. Ils utilisent essentiellement trois opérateurs de base Sépare, Attache et Itération, complétés par des opérateurs de liens avec l'environnement. Leur fonctionnement est conditionné par la présence des données à traiter et par des opérateurs de contrôle modélisés par RdPFC. Il y a ainsi interaction permanente entre les deux réseaux.

Pour chacun de ces opérateurs, on donne dans cet article, leur traduction matérielle sous forme d'une

bibliothèque de composants VHDL. Plusieurs exemples simples d'application sont décrits. Le plus complet, donné en fin d'article, concerne la robotique mobile d'assistance au handicap (Ben Khelifa, 2001).

Pour évaluer les capacités fonctionnelles d'une personne handicapée motrice et les compensations nécessaires à la commande d'un fauteuil roulant électrique, nous avons réalisé une plateforme de test appelée FRACAH (Fauteuil Roulant A Commande Adaptée au Handicap). Ce fauteuil roulant peut être équipé de différentes commandes modulaires adaptées à la personne handicapée. On peut ainsi le commander par la main, les pieds, la tête, la voix et le souffle. Par exemple, la commande manuelle par joystick est gérée par un réseau de neurones qui apprend les limitations fonctionnelles de la main et compense tous les gestes incomplets. Il est bien évidemment équipé de plusieurs capteurs de sécurité.

2 Les Réseaux de Petri Architecturaux

Ils ont été définis pour offrir une solution au problème de l'Adéquation Algorithme Architecture. En effet, l'étude de la conception conjointe matérielle/logicielle doit être conduite avec beaucoup d'attention dans la perspective de l'obtention d'une solution optimisée et réalisant le meilleur compromis quantité de ressources/temps de traitement/coût. Pour y parvenir, un modèle de représentation du parallélisme intrinsèque d'un algorithme est indispensable. Les Réseaux de Petri à Flux de Données constituent une très bonne base d'études. Ils ont été utilisés comme outil de modélisation d'un méthode d'Adéquation définie par A.F. Dias (Dias, 2000). Il en existe deux types :

- Les Réseaux de Petri à Flux de Données Défactorisés (RdPFDD) pour l'utilisation du parallélisme maximal sur une architecture non bornée,
- Les Réseaux de Petri à Flux de Données Factorisés (RdPFDF) pour l'utilisation d'un parallélisme adapté à la quantité de ressources bornée d'une architecture existante.

2.1 Réseaux de Petri à Flux de Données Défactorisés

2.1.1. Représentation

Ils ont été définis à partir d'un modèle élaboré par J. Alhmana (Alhmana, 1983) et reposent essentiellement sur la notion de places biparties (opérateurs et variables) (figure 1).

2.1.2. Définition

Un Réseau de Petri à Flux de Données (RdPFD) est un septuplet (R, φ, ξ, ψ, X, O, C) défini de la façon suivante :

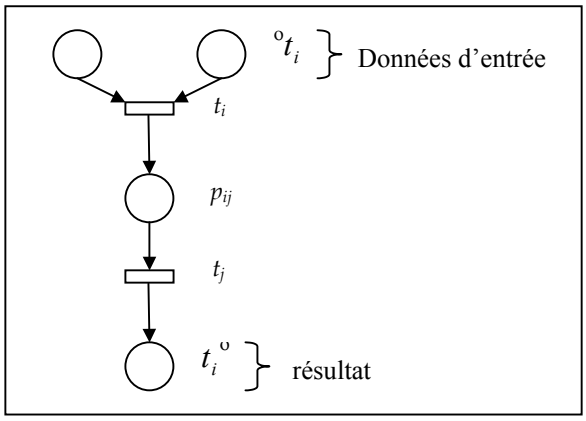


Figure 1. *Réseau de Petri à Flux de Données*

- R est un RdP-PbP conforme,
- φ est une application : φ | $P_V : P \rightarrow X$
  φ | $P_O : P \rightarrow O$

telle que

$$\forall p_i \in P_O, \forall p_j \in P_O \text{ et } \varphi(p_i) = \varphi(p_j) \text{ pour } i \neq j,$$
$$\text{alors } \forall t_l \in {}^\circ p_l \text{ et } \forall t_k \in {}^\circ p_k, \varphi({}^\circ t_l) = \varphi({}^\circ t_k).$$

Ainsi, deux opérateurs identiques ne peuvent pas travailler sur le même ensemble de données.

- ξ est une application injective $\xi : X \rightarrow ME=\{ME_1, \ldots, ME_n\}$ telle que

$$\forall p \in P_V, \exists ME_i \in ME \mid ME_i = \xi(\varphi(p)).$$

ME est appelé ensemble de zones mémoires.

- ψ est une application qui permet de poser des conditions sur le franchissement d'une transition $\psi : Tr \rightarrow C$.
- X = $\{x_1,x_2,...,x_s\}$ est un ensemble de variables réelles, entières, logiques… prenant leurs valeurs respectivement dans les domaines D1, D2, … Du.
- O = $\{o_1,o_2,...,o_v\}$ est un ensemble fini d'opérateurs définis comme des applications internes de D1 × D2 × … × Du.
- C = $\{c_1, c_2, \ldots c_r\}$ est un ensemble de prédicats sur les variables de X.

2.1.3. Marquage

Soient R un RdPFD et M_0 son marquage initial. Le graphe des marquages conséquents Q' est le graphe orienté (S,L) dans lequel $S = M_0$ est l'ensemble des sommets et L est l'ensemble des arcs (M_i, M_j) tels que :

$$\forall M_i \in M_0, \forall M_j \in M_0, \exists t_i \text{ telle que } M_i \xrightarrow{t_i} M_j.$$

Les arcs du graphe Q' sont étiquetés par les transitions du réseau. La figure 2 montre le RdPFD correct et le graphe des marquages associés au calcul d'un terme de produit matriciel $a_1 = w_{11}.V_1 + w_{12}.V_2$

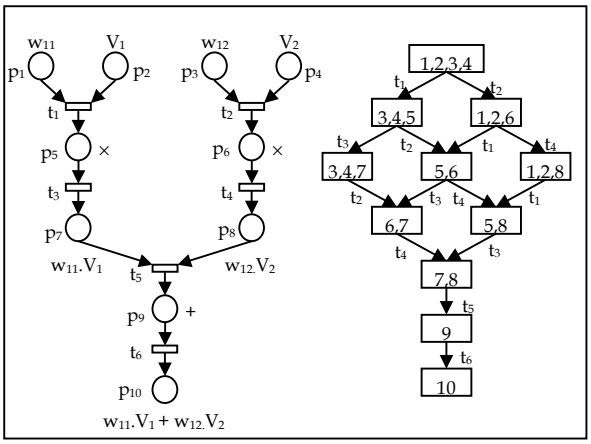


Figure 2. *RdPFD et graphe des marquages*

2.1.4. Opérateurs

La nature des données à traiter pouvant être très variée (entier, logique, vecteurs, matrices…), les opérateurs peuvent réaliser différentes fonctions. En plus des opérateurs de calcul tels ceux de la figure 2, on trouve également des opérateurs de composition et de décomposition.

Opérateur Compose : Il est identifié par Co et il effectue le regroupement ordonné de d données d'entrées de même type T'$_1$ à T'$_d$, en un vecteur de sortie de dimension d, [T'_1 ... T'_d]. (figure 3).

$$\text{Co}(\ T'_1\ ,\ \dots\ ,\ T'_d\ ) \rightarrow [T'_1\ \dots\ T'_d]$$

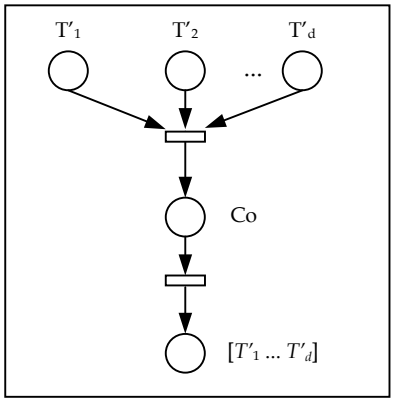


Figure 3. *RdPFD de l'opérateur Compose*

Opérateur Décompose : Il est identifié par De, et il effectue la décomposition d'un vecteur [T'_1 ... T'_d] en ses éléments T'_1 à T'_d (figure 4). C'est l'inverse de l'opérateur Compose.

$$\text{De}([T'_1\ \dots\ T'_d]) \rightarrow T'_1\ ,\ \dots\ ,\ T'_d$$

2.1.5. Exemple

Pour illustrer une application des définitions précédentes, considérons un filtrage classique, lequel est utilisé pour le traitement des images de la webcam du fauteuil roulant qui sera présenté au paragraphe 3.

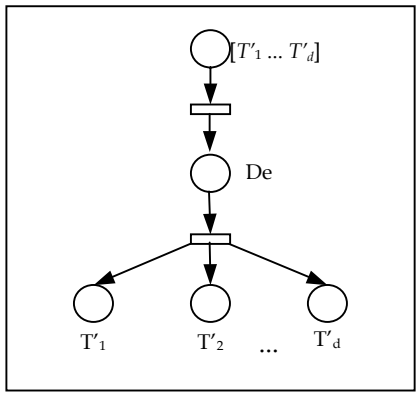


Figure 4. *RdPFD de l'opérateur Décompose*

Il consiste à appliquer au pixel central d'un motif répétitif horizontalement et verticalement l'opération : $\varphi : p_0^{\ nouveau} = \varphi(p_0, p_1, \dots, p_N)$. Dans le cas d'un filtre linéaire 3×3, la nouvelle valeur de p_0 est calculée par :

$$p_0^{\ nouveau} = \frac{1}{j}\sum_{i=0}^{8} j_i . k_i$$

k_i étant les valeurs du motif répétitif :

$$\frac{1}{S_k}.\begin{vmatrix} k_0 & k_1 & k_2 \\ k_3 & k_4 & k_5 \\ k_6 & k_7 & k_8 \end{vmatrix}\ \text{, soit}\ \frac{1}{16}.\begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix}$$

pour l'exemple du fauteuil roulant de cet article. L'application de cette opération sur une image m×n conduit à l'obtention d'une image (m-2)×(n-2). Le RdPFDD du filtrage est donné par la figure 5.

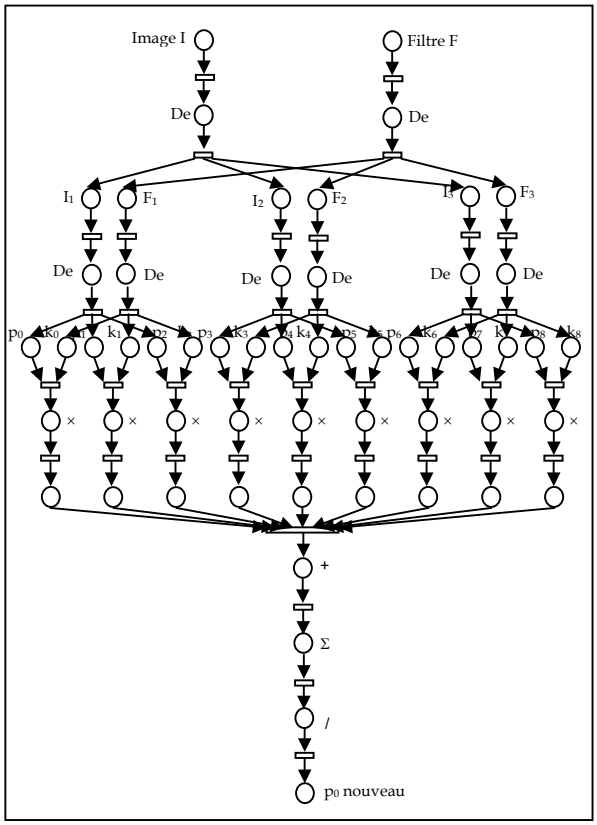


Figure 5. *RdPFDD du filtrage 3×3*

Cette version entièrement parallélisée est facilement programmée en VHDL pour une implantation automatique sur composant programmable type FPGA. Elle permet d'obtenir le temps de traitement le plus court pour une utilisation de tous les opérateurs fonctionnant en parallèle. Ces réseaux peuvent être décrits facilement en VHDL en procédant comme le montrent la figure 6 pour chaque opérateur et la figure 7 pour l'association des opérateurs (Abellard & al., 2004).

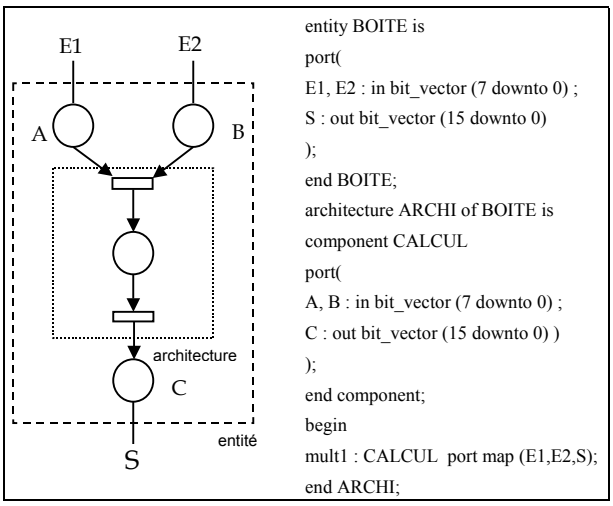


Figure 6. *Description VHDL d'un opérateur à partir de RdPFD*

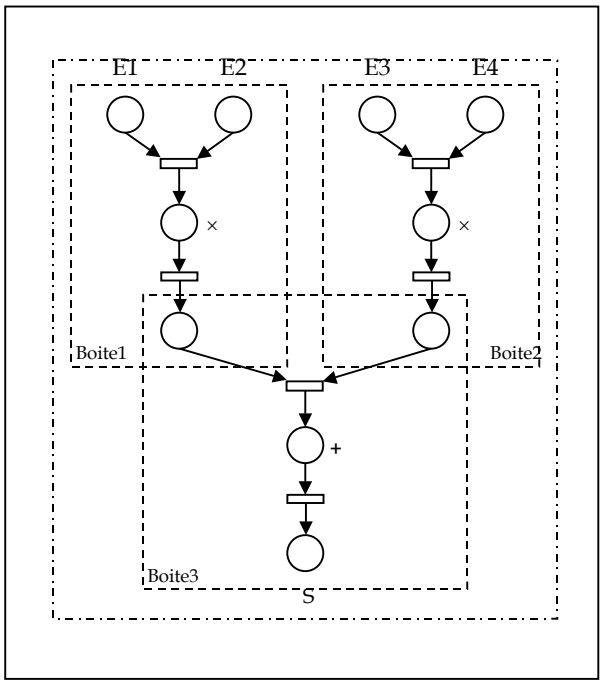


Figure 7. *Principe de l'association de plusieurs opérateurs*

Par contre, quand le nombre d'opérateurs utilisés augmente avec la complexité des calculs et/ou que le composant programmable n'est utilisable qu'en partie (multitâches), il peut être intéressant de profiter de la vitesse d'exécution des FPGA en utilisant un double traitement parallèle et séquentiel des données, surtout dans le cas d'opérations répétitives.

2.2 Réseau de Petri à Flux de Données Factorisé

La factorisation des opérations conduit à grouper de manière ordonnée les données d'un même type. Les types représentent le codage binaire des données.

2.2.1. Définition

Un RdPFDF est un doublet (*R*, *FF*) dans lequel R est un RdPFD et FF un ensemble de frontières de factorisation, internes et externes : $FF=\{FF_1,FF_2,\ldots,FF_n\}$.

2.2.2. Opérateurs

Opérateur Sépare : Il est identifié par Se, et il effectue la factorisation d'un flux de données, sous la forme d'un vecteur $[T'_1 \ldots T'_d]$ en entrée, en énumérant les éléments T'_i en sortie, $i \in \{1, \ldots, d\}$. Un changement de la valeur des données en entrée de l'opérateur Sépare correspond à d changements de la valeur des données en sortie.

$$Se([T'_1 \ldots T'_d]) \rightarrow T'_i$$

L'opérateur Sépare permet de traverser une frontière de factorisation en augmentant la cadence des données : en aval de Sépare, la cadence des données est d fois supérieure à la cadence des données en amont. A une donnée d'entrée (côté lent) correspondent d données en sortie (côté rapide) résultant de l'énumération des éléments de la donnée d'entrée. Ainsi, une frontière de factorisation FF définie par un opérateur Sépare dissocie le côté « lent » et le côté « rapide ». (figure 8)

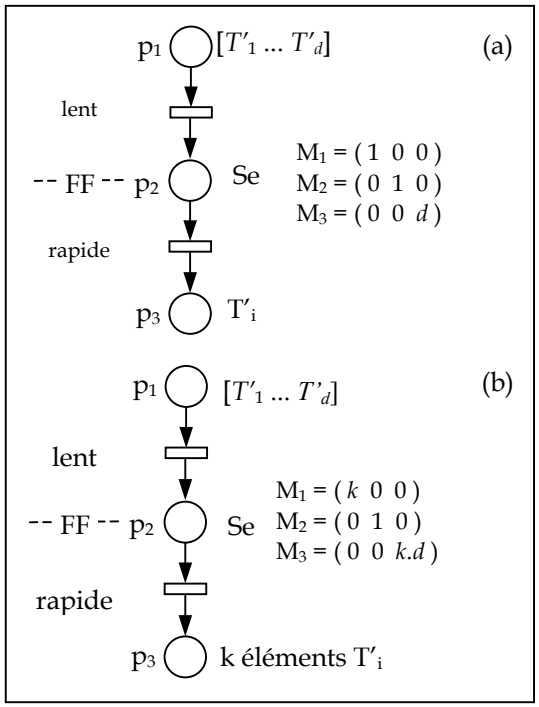


Figure 8. *RdPFDF de l'opérateur Sépare*
(a) : Principe de fonctionnement pour une donnée d'entrée comprenant d éléments
(b) : Généralisation pour la séparation de k données d'entrée ayant d éléments chacune

Opérateur Attache : Il est identifié par At et il effectue la factorisation des d flots de données T'i en entrée, i = {1, ..., d} en les collectant sur la forme d'un vecteur [T'_1 ... T'_d] en sortie.

$$At(T'_i) \rightarrow [T'_1 \ldots T'_d]$$

- d changements de la valeur des données en entrée de l'opérateur Attache correspondent à un changement de la valeur de la donnée en sortie
- l'opérateur Attache permet de traverser une frontière de factorisation en réduisant la cadence des données. En amont de l'opérateur Attache, la cadence des données est d fois supérieure à la cadence des données en aval.

La frontière de factorisation FF de l'opérateur Attache sépare le côté « lent » du côté « rapide » (figure 9).

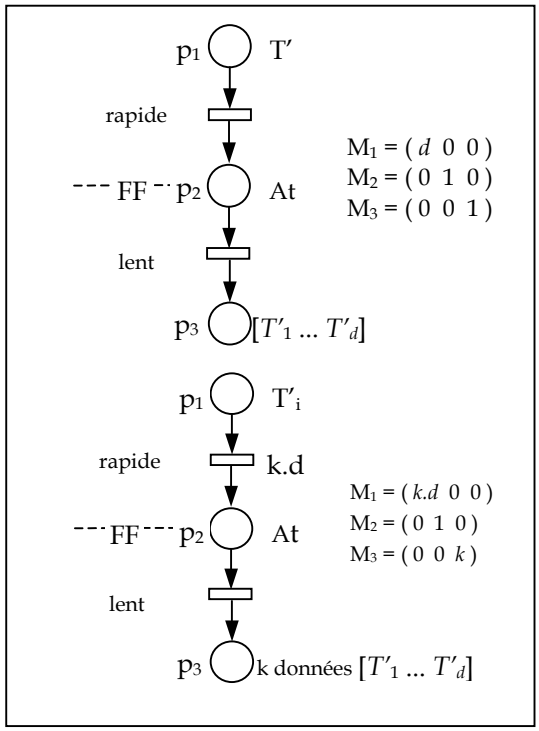


Figure 9. *RdPFDF de l'opérateur Attache*
(a) : Principe de fonctionnement pour l'attachement de d éléments T'_i en une donnée [T'_1 ... T'_d]
(b) : Généralisation pour l'attachement d'éléments T'_i en k données ayant chacune d éléments

Opérateur Itération : Il est identifié par It et il effectue la factorisation des dépendances de données entre les réseaux. Lorsque d sous-réseaux SR identiques sont factorisés, chaque dépendance de données entre deux sous-réseaux apparaît en entrée et en sortie du sous-réseau factorisé. L'opérateur Itération permet de marquer chaque dépendance de données inter-sous-réseaux, afin que le cycle apparent soit marqué comme n'étant pas un cycle de dépendance.

L'entrée e reçoit les données qui arrivent du sous-réseau et la sortie s fournit les données d'entrée du sous-réseau. L'opérateur Itération permet de spécifier une connexion entre sous-réseaux répétitifs et apparaît dans le RdPFD factorisé comme un cycle à travers un opérateur It (figure 10). Sur la figure, i∈{1…d} est un signal issu d'un compteur en sortie d'un opérateur de contrôle.

$$It(init, e) \rightarrow (s, fin)$$

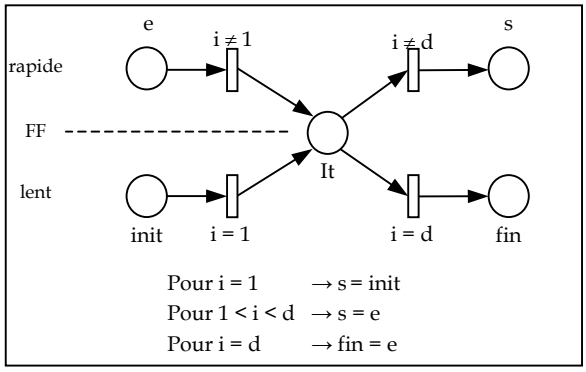


Figure 10. *RdPFD de l'opérateur Itération*

2.2.3. Exemple

En reprenant l'exemple du filtrage précédent, le RdPFDF correspondant est donné sur la figure 11. La factorisation permet de s'affranchir de la limitation physique liée au nombre de cellules disponibles dans le FPGA puisque les données sont traitées séquentiellement par énumération. Les places de validation viennent du réseau de contrôle.

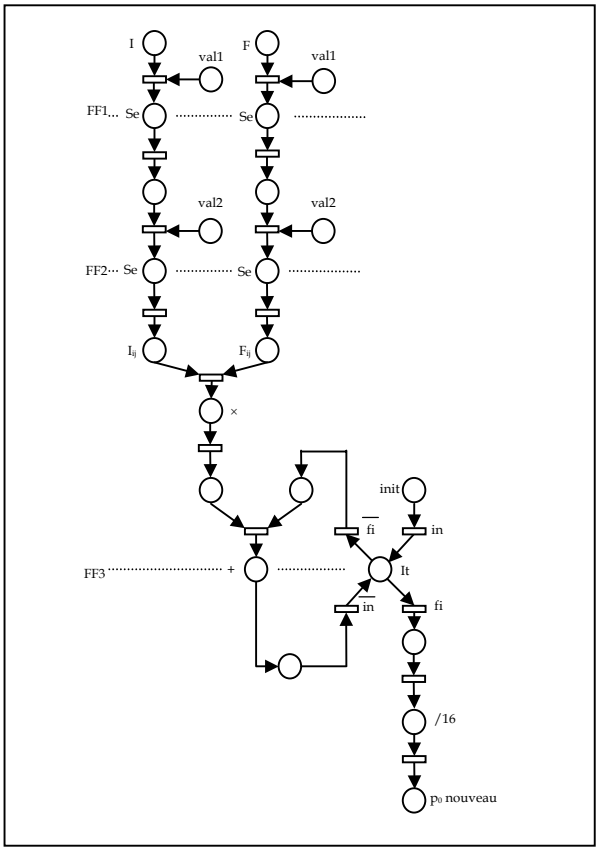


Figure 11. *RdPFDF du filtrage 3×3*

2.3 Réseaux de Petri à Flux de Contrôle

La génération du contrôle d'une implantation matérielle à partir de la spécification algorithmique, c'est-à-dire du RdPFDF est déduite des relations de production et de consommation de données et des relations de voisinage entre les FF. La synthèse du contrôle d'une implantation matérielle consiste à générer, à partir des dépendances de données entre les opérateurs FF, les signaux de validation et d'initialisation qui commandent les compteurs. Ceux-ci génèrent les signaux de validation des registres associés aux opérateurs. Ainsi, la génération de la partie contrôle de l'implantation matérielle correspondant à la spécification algorithmique de l'application décrite par RdPFD est modélisée par Réseau de Petri à Flux de Contrôle (RdPFC).

2.3.1. Définition

Un RdPFC est un triplet (R, FF, P_c) dans lequel :
- R est un Réseau de Petri à Places bi-Parties.
- FF est l'ensemble des frontières de factorisation
- P_c est l'ensemble des places de contrôle.

2.3.2. Synthèse de contrôle

Elle est effectuée en 5 étapes :
- établissement du RdPFDF
- élaboration du RdP de voisinage entre frontières
- à partir de ce RdP, définition des relations de voisinage, de production et de consommation entre les frontières
- génération des équations des signaux de contrôle
- modélisation par RdPFC en interconnectant les unités de contrôle associées à chaque FF en fonction des équations des signaux de contrôle.

2.3.3. Unité de contrôle

Dans un circuit séquentiel, c'est-à-dire comprenant des registres, chaque FF a des relations amont et aval des deux côtés (lent et rapide). Les relations entre les signaux de demande et acquittement, en amont et aval, des deux côtés lent et rapide constituent l'unité de contrôle de la FF. L'unité de contrôle est constituée d'un compteur à D états et d'une logique supplémentaire qui génère :

- le protocole de communication
- la valeur cpt pour les opérateurs frontières internes
- le signal de validation val qui commande les opérateurs frontières externes.

La modélisation d'une unité de contrôle est donnée en figure 12. Les abréviations utilisées sont :

- d_m : demande amont
- d_v : demande aval
- a_m : acquittement amont
- a_v : acquittement aval

Règles de fonctionnement : si OC reçoit une demande en amont (d_m = 1) et que l'acquittement en aval est terminé (a_v = 0), alors OC valide le transfert des données (a_m = 1) et envoie une demande à l'opérateur suivant (d_v = 1). Si une nouvelle demande d_m est présente alors que a_v n'est pas encore activé, OC ne valide pas un nouveau transfert de données qui est alors mis en attente.

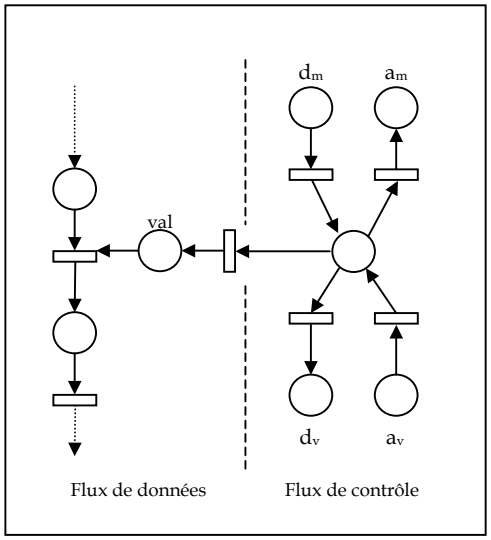


Figure 12. *Modélisation d'une unité de contrôle*

Le RdPFC correspondant au RdPFDF de la figure 11 est donné sur la figure 13.

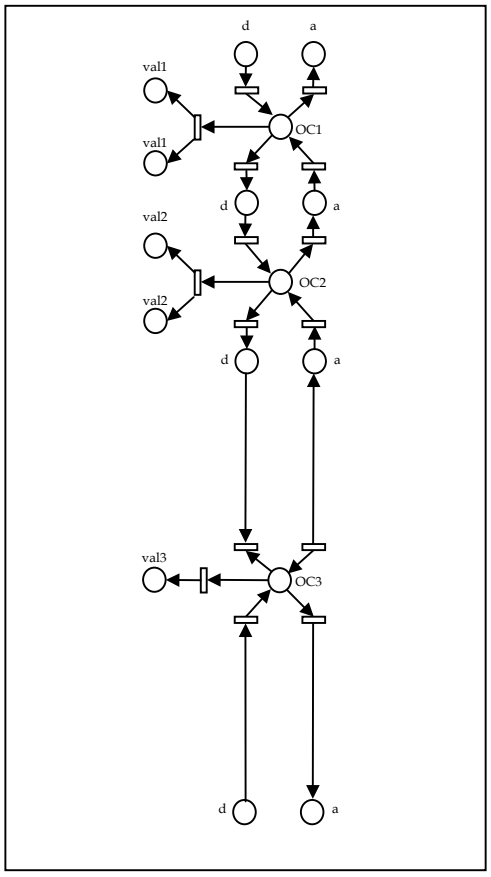


Figure 13. *RdPFC du filtrage 3×3*

2.4 Réseaux de Petri Architecturaux

Ils résultent de l'unification en un modèle des RdPFDF précédents et des RdPF de Contrôle qui permettent la commande de données (figure 14). Les RdPFC sont constitués d'opérateurs de contrôle fonctionnant par procédure demande/acquittement. Un exemple est donné sur la figure 13. C'est donc le modèle le plus concis qu'il est possible d'obtenir et qui permet d'obtenir la quantité de ressources minimales quelle que soit la taille du filtre.

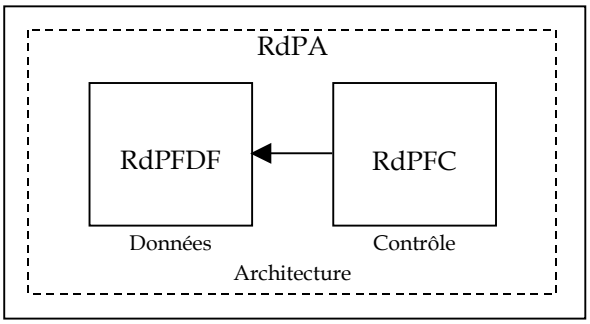


Figure 14. *Unification des modèles flux de données et flux de contrôle en RdPA*

La figure 15 nous montre le RdPA complet du filtrage linéaire, par unification des RdP des figures 11 et 13.

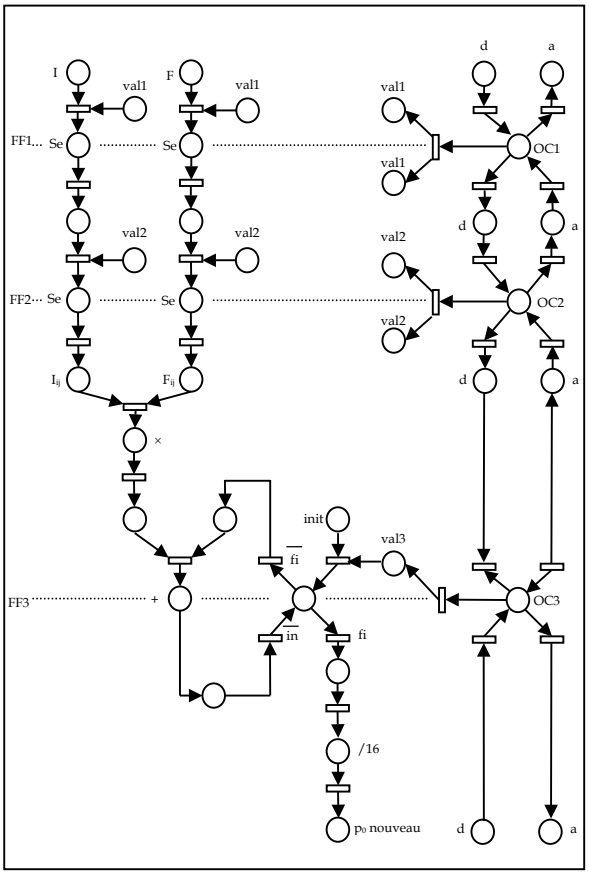


Figure 15. *RdPA du filtrage linéaire*

3. Robotique mobile d'assistance au handicap

Le fauteuil roulant électrique est un élément important pour l'autonomie des personnes handicapées à mobilité réduite. Pour un grand nombre d'entre elles, son utilisation peut toutefois être difficile, même impossible parfois à cause par exemple de capacités physiques résiduelles trop faibles, d'une fatigabilité importante ou d'une trop grande spasticité (contractures musculaires, perturbations des réflexes et du tonus des muscles pouvant entraîner tremblements saccadés ou hypertonie brusque) (Bourhis & al., 2000). C'est ainsi que certaines personnes ne peuvent actionner qu'un capteur d'interface homme-machine tout ou rien, ce qui rend difficile, voire même impossible la commande du fauteuil dans le cas de manœuvres précises. Dans ce cas, l'une des rares possibilités de commande se réduit à la sélection sur un écran ou sur un défileur à LED d'une direction avant (AV)-arrière (AR)-droite (DR)-gauche (GA). Lorsqu'une simple amélioration de l'interface homme-machine standard de type joystick ne suffit pas à palier les difficultés de la commande, il est tout à fait possible d'ajouter des asservissements locaux ou des fonctionnalités intelligentes identiques à celles utilisées en robotique mobile manufacturière.

La robotique de réadaptation et plus précisément l'étude des fauteuils roulants intelligents, s'apparente au domaine de la robotique de service. Il en découle deux caractéristiques importantes :

- le fauteuil partage l'espace d'évolution avec les individus,
- l'utilisateur n'est pas a priori compétent dans le domaine technologique.

Ceci met en évidence l'importance de l'interface homme-machine. Par ailleurs, un fauteuil roulant évolue en général en intérieur, donc dans un espace structuré par des balises naturelles (murs, portes...). Mais il arrive que, en situation réelle, des modifications d'un environnement préalablement modélisé peuvent apparaître (porte fermée, objets ou personnes gênant ou interdisant un passage...). Les problèmes de coopération homme-machine sont donc particulièrement sensibles.

La grande diversité des utilisateurs et des situations est une caractéristique importante du problème à résoudre. Par exemple, l'environnement peut être le domicile de l'utilisateur ou une structure comme un centre de réadaptation ou un hôpital. De fait, l'encombrement potentiel des chemins et les distances à parcourir sont donc très variables.

Par ailleurs, les pilotes de fauteuil ont également des possibilités physiques et cognitives très variées. Il est essentiel, dans une application d'aide technique pour personnes handicapées, de prendre en considération le rapport coût/performance dans les choix technologiques, à cause de l'étroitesse du marché potentiel actuel (Fuhrer, 1997). Il est ainsi

préférable d'utiliser, chaque fois que c'est possible, du matériel non spécifique à l'application. Par exemple, sauf si l'on veut apporter une fonctionnalité mécanique supplémentaire comme une caractéristique omnidirectionnelle, il faut utiliser comme base mobile, un fauteuil électrique du commerce sur lequel on viendra réaliser toutes les adaptations nécessaires.

Le coût conditionne également beaucoup le choix des capteurs. C'est ainsi que les capteurs ultrasoniques sont souvent utilisés, parfois en association avec des capteurs infrarouges. Pour la réalisation de fonctionnalités assez simples, il est également envisageable d'utiliser des caméras lorsqu'il n'y a pas besoin de mettre en oeuvre des traitements informatiques lourds. L'implantation des capteurs doit être discrète, et le design et le confort du fauteuil soigné pour éviter des problèmes d'acceptabilités psychologiques de l'utilisateur.

L'interface homme-machine doit être très conviviale car elle est destinée à des utilisateurs non-spécialistes des problèmes technologiques, et le dialogue ne doit porter que sur un nombre réduit de commandes pour les personnes aux capacités physiques résiduelles réduites.

Les déplacements automatiques du fauteuil doivent apparaître naturels pour l'utilisateur, c'est-à-dire que le parcours autonome doit correspondre à celui qu'il aurait utilisé avec une commande manuelle directe... Les trajectoires doivent donc être lissées et les changements de directions prédictibles. Dans tous les cas, l'utilisateur doit rester constamment maître du contrôle, c'est-à-dire qu'il doit toujours pouvoir intervenir en cas de problèmes, en stoppant le fauteuil.

Deux aspects sont particulièrement sensibles dans cette application : la robustesse et la sécurité, car l'utilisateur est placé sur le fauteuil et il a des capacités physiques réduites. A tout moment, l'utilisateur doit pouvoir reprendre le contrôle, même dans une phase de déplacement automatisé si un problème lié à la sécurité apparaît, par une action unique sur le capteur de commande.

Cette commande de sécurité doit être adaptée à chaque utilisateur en tenant compte des limitations liées à son handicap. Plusieurs auteurs ont montré que même dans des réalisations particulièrement soignées, il était difficile d'obtenir une sécurité et une fiabilité totales (Van der Loos, 1992). Il est donc important de bien spécifier quel rapport coût-risque est acceptable en fonction du bénéfice espéré.

Du fait de la grande variété des handicaps, l'interface homme-machine d'un fauteuil robotisé doit être modulaire et facilement configurable, aussi bien du point de vue logiciel que matériel (capteurs de commande). Il en existe une grande diversité et peuvent être classés en trois catégories du point de vue fonctionnel (Roy & al., 1992) :

- le capteur proportionnel est en général un capteur de type joystick commandé manuellement ou plus rarement par le menton, la nuque ou les pieds (Ferrario, 1992),
- le capteur tout ou rien est un simple interrupteur commandé par exemple par des appuis de tête. Il est alors nécessaire, pour un contrôle complexe de l'environnement, de l'associer à un système de balayage de choix sur un écran ou sur une matrice à LEDs,
- l'utilisation de plusieurs capteurs tout ou rien permet la combinaison d'une commande d'action avec une commande d'arrêt de sécurité (capteurs de souffle, reconnaissance vocale...).

En cas de nécessité, toutes les données provenant de ces capteurs peuvent être filtrées, par exemple pour des personnes souffrant de tremblements de la main (Hendriks & al., 1991). L'interface graphique permettant le dialogue homme-machine dépend du capteur utilisé. Dans le cas d'un unique capteur tout ou rien, il est primordial d'optimiser le temps d'accès aux commandes. En effet, pour certaines personnes handicapées, le temps minimum entre deux actions sur un capteur peut atteindre plusieurs secondes. La solution consiste généralement à organiser l'ensemble des icônes en différents écrans d'autant plus rapidement accessibles que la fréquence de leur choix est élevée.

C'est pour prendre en compte la diversité de ces handicaps que le projet FRACAH a été initié.

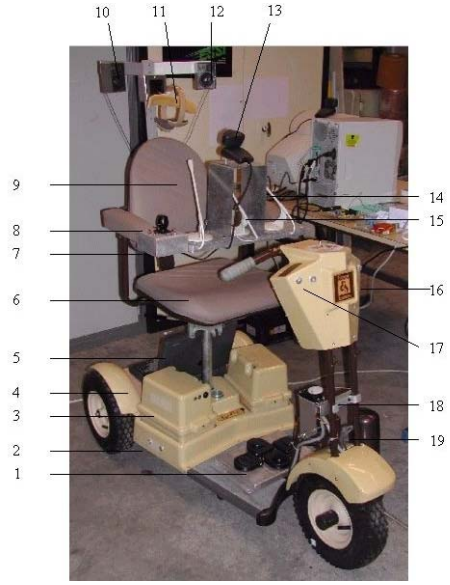
4. Le FRACAH

Pour évaluer les capacités opératoires d'une personne handicapée et définir les compensations nécessaires à la commande d'un fauteuil roulant électrique, une plateforme technologique a été réalisée à partir de matériel standard et de modules interchangeables. Cette plateforme a été appelée FRACAH (Fauteuil Roulant à Commande Adaptée au Handicap) (figure 16).

Les deux commandes de direction et de propulsion peuvent être adaptées à chaque personne en fonction de ses capacités résiduelles (figure 17).

Lorsque la commande la plus appropriée à la personne handicapée a été choisie, une phase d'éducation et d'apprentissage de la commande du fauteuil est mise en place. Par exemple, la commande par la main est faite par un réseau de neurones qui apprend les limitations fonctionnelles et compensent les gestes incomplets (Abellard & al., 2004) (figure 18).

La sécurité est assurée par des capteurs de proximité et une commande à distance (voie hertzienne) par le surveillant qui contrôle les phases d'apprentissage et peut prendre le contrôle du fauteuil lorsque le patient est en difficulté, entre autres par une webcam qui surveille les yeux du patient. Si celui-ci a les yeux ouverts, le fauteuil peut fonctionner ; si les yeux sont baissés ou fermés (ce qui est fréquent losque le patient regarde les commandes au début de la phase de pilotage), le fauteuil est arrêté.



1. Commande par pédales
2. Détecteurs de proximité droite
3. Compartiment batteries
4. Compartiment cartes électroniques
5. Micro-ordinateur
6. Siège motorisé en rotation et élévation
7. Commande par manette neuronale
8. Voyants-guides directionnels
9. Commande par le souffle (avant/droite)
10. Haut-parleur droit (synthèse vocale)
11. Commande par la tête
12. Haut-parleur gauche (synthèse vocale)
13. Caméra (mouvements tête, yeux…)
14. Commande par le souffle (arrière/gauche)
15. Microphone pour commande vocale
16. Détecteur de proximité avant gauche
17. Détecteur de proximité avant droit
18. Servomoteur de direction
19. Moteur de propulsion

Figure 16. *Le FRACAH (photo)*

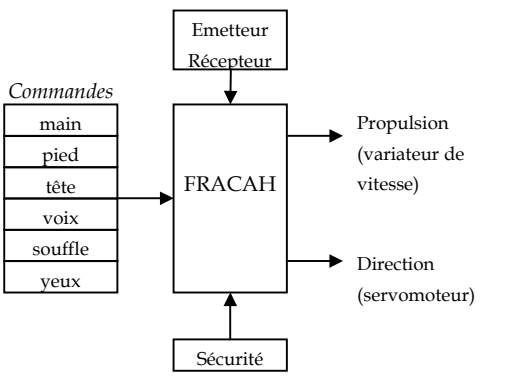


Figure 17. *Schéma fonctionnel du FRACAH*

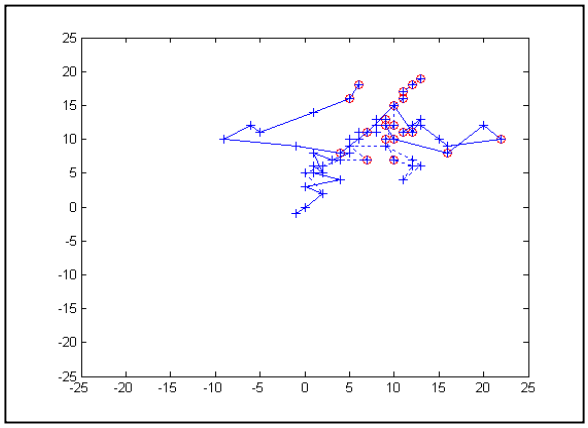


Figure 18. *Exemple d'acquisition et de reconnaissance des gestes par le réseau de neurons (pour la direction "avant").*

C'est pour cette fonction de sécurité qu'a été réalisé le filtrage présenté au début de l'article. Le traitement d'images complet repose sur 5 phases :

- un prétraitement par filtrage linéaire pour éliminer le bruit,
- une binarisation,
- une extraction de contours pour repérer le blanc de l'œil ouvert,
- une segmentation en régions pour déterminer la surface du blanc de l'œil,
- une comparaison par rapport à un seuil fixé afin de déterminer la situation : « œil ouvert » ou « œil fermé ». C'est cette dernière information binaire qui conditionnera le fonctionnement ou l'arrêt du moteur (figure 19).

  - œil ouvert : moteur en marche,
  - œil fermé : moteur arrêté.

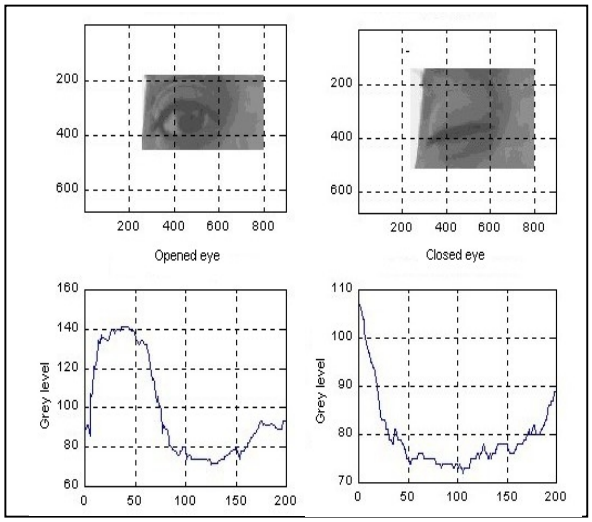


Figure 19. *Traitement de l'image des yeux*

Tous ces opérateurs ont été implantés dans un composant FPGA à partir d'un programme VHDL

résultant de la modélisation des algorithmes par RdPA. Toutes les commandes ont été réalisées sur le même principe en temps réel.

Les résultats obtenus en simulation avec un composant Stratix (Altera), sont résumés dans le tableau de la figure 20.

	V. défactorisée		V. factorisée	
	LE	temps	LE	temps
1 point	1212	73 ns	391	210 ns
image 200×200	″	2,9 µs	″	8,4 µs

Figure 20. *Caractéristiques de l'implantation de l'opérateur de filtrage. Comparatif des versions factorisées et défactorisées suivant le nombre d'éléments logiques (LE) et le temps d'exécution.*

Quand on compare la version factorisée à la description défactorisée, le gain en termes de cellules logiques (matériel) est non négligeable, tandis que l'allongement du temps d'exécution n'est pas dommageable pour un traitement rapide. Ainsi, les éléments logiques inutilisés pourront servir à d'autres calculs et traitement de signaux parallèles (notamment ceux provenant des capteurs de sécurité).

5. Conclusion

Les RdPA constituent un outil performant d'analyse, de simulation et de modélisation d'algorithmes parallélisables. Ils conduisent à l'obtention automatique d'un programme VHDL pouvant être implantés sur des composants programmables. Ils peuvent s'appliquer à de nombreuses applications temps réel. L'exemple décrit dans cet article traite d'un domaine étudié au laboratoire et concerne l'assistance technique au handicap.

Le système de commande modulaire mise au point est intéressant car il peut s'adapter à tout type de fauteuil roulant du commerce et au handicap du patient. D'une manière générale, c'est le matériel qui s'adapte à la personne handicapée et non pas le contraire.

Remerciements

Nous remercions le Conseil Régional Provence-Alpes-Côte d'Azur pour son soutien financier, et M. J.P. Belheur, ergothérapeute, qui est à l'origine d'un projet de Mobile Autonome Robotisé pour Handicapés, dont le FRACAH peut être considéré comme étant une première étude de faisabilité.

Références

Abellard A., Abellard P., Grall P., Razafindrakoto N., Ben Khelifa M. : A Man-Machine Interface based on a Petri Net Neural Control for Mobile Robotics Applied to Handicapped Persons, *I3M AVCS*, pp. 41-47, Genova, 2004.

Almhana J. : Modélisation par réseaux de Petri de Données. Application à la synthèse de l'opérateur de Riccati rapide. Thèse de doctorat, Université Aix-Marseille III, 1983.

Ben Khelifa M. : Vision par ordinateur et robotique d'assistance : Application au projet MARH, Mobile Autonome Robotisé pour Handicapés. Thèse de doctorat, Université de Toulon, 2001.

Bourhis G., Gelin R., Pruski A. : *Robotique d'aide aux personnes handicapées, in Applications non manufacturières de la robotique* (P. Dauchez), pp. 193-238, Ed. Hermès, 2000.

Dias A.F. : Contribution à l'implantation optimisée d'algorithmes bas niveau de traitement du signal et des images sur des architectures mono-FPGA à l'aide d'une méthodologie d'Adéquation Algorithmes Architectures. Thèse de doctorat, Université Parix XI Orsay, 2000.

Diaz M. : *Les réseaux de Petri. Modèles fondamentaux. Informatique et Systèmes d'Information, Commande, Communication*. Ed. Hermès, Science Publications, 2001.

Ferrario M., Lodola M. : A multifunctional nape joystick. *RESNA 15th Annual Conference*, pp. 524-526, Toronto, Canada, 1992.

Fuhrer M. : *Evaluating the outcomes of assistive technologies and related services. The role of program management-oriented and rehabilitation science-oriented outcomes research. Advancement of Assistive Technology*, pp. 44-47, IOS Press, 1997.

Hendriks J.L., Rosen M.J. : A second-generation joystick for people disabled by tremor. *RESNA 14th Annual Conference*, pp. 248-250, 1991.

Roy D.M., Panayi M. : Advanced input method for people with Cerebral Palsy : a vision to the future. *RESNA 15th Annual Conference*, pp. 99-100, Toronto, Canada, 1992.

Van Der Loos H.F.M. : Safety considerations for rehabilitative and human-service robot systems. *RESNA 15th Annual Conference*, pp. 321-324, Toronto, Canada, 1992.